Computational Aeroelastic Analysis of Micro Air Vehicle with Experimentally Determined Modes

THESIS

Joshua Allen Stults, Second Lieutenant, USAF

AFIT/GAE/ENY/05-M23

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/05-M23

Computational Aeroelastic Analysis of Micro Air Vehicle with Experimentally Determined Modes

THESIS

Presented to the Faculty

Department of Aeronautical and Astronautical Engineering

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aeronautical Engineering

Joshua Allen Stults, B.S.

Second Lieutenant, USAF

March, 2005

AFIT/GAE/ENY/05-M23

Computational Aeroelastic Analysis of Micro Air Vehicle with Experimentally Determined Modes

Joshua Allen Stults, B.S.

Second Lieutenant, USAF

Approved:

/signed/	21 March 2005
LtCol Raymond C. Maple (Chairman)	date
/signed/	21 March 2005
Dr Richard G. Cobb (Member)	date
/signed/	21 March 2005
Dr Mark F. Reeder (Member)	date

AFIT/GAE/ENY/05-M23

Abstract

A computational aeroelastic analysis of a micro air vehicle (MAV) is conducted. This MAV has a 24 inch wing span, and is designed for local area reconnaissance. Wind tunnel data for the MAV with a rigid carbon fiber wing and a flexible carbon fiber ribbed nylon wing are compared to CFD results incorporating static and dynamic deformations. We use laser vibrometry to determine the mode shapes of the flexible wing. From these shapes, CFD grid deformations are calculated as part of a closely-coupled aeroelastic solution method. The accuracy of MAV performance predictions using CFD with and without aeroelastic modeling is evaluated against previous wind-tunnel experiments. The performance benefits of the flexible wing, and the applicability and limitations of the model are evaluated in the present research effort. Some suggestions are made as to improvements that can be made to increase the range of applicability and the accuracy of the model.

Acknowledgements

This effort owes its completion in a large part to the work, help, and advice of Maj Gregory H. Parker, Dr Richard G. Cobb, and, far from least of all, my advisor LtCol Raymond C. Maple. I am very grateful for their guidance and patience, which helped me to learn a great deal from this experience, and hopefully produce a useful piece of research as well.

With regards to my fellows at the institute, who were always up for providing much needed distraction and even scholarly advice on occasion, I have nothing but best wishes and heartfelt thanks.

My wonderful fiance who put up with me being terribly anti-social for extended periods of time for the duration of this thesis, deserves more than written praise.

Joshua Allen Stults

Table of Contents

Page

List of Figures

List of Tables

Nomenclature

α	Angle of Attack, degrees
$\ddot{d}$	vector of nodal accelerations
$\ddot{z}$	vector of modal accelerations
$\dot{d}$	vector of nodal velocities
$\dot{z}$	vector of modal velocities
ω	Natural frequency, rad/s
Φ	Modal matrix
d	vector of nodal displacements
f	Natural frequency, Hz
K	Stiffness matrix
M	Mass matrix
z	vector of modal displacements
CFD	Computational Fluid Dynamics
CSD	Computational Structural Dynamics
DES	Detached Eddy Simulation
ISR	Intelligence, Surveillance, and Reconnaissance
MAV	Micro Air Vehicle
PISO	Pressure Implicit with Splitting Operators

Computational Aeroelastic Analysis of Micro Air Vehicle with Experimentally Determined Modes

I. Introduction

Micro air vehicles (MAVs) are small unmanned aircraft developed for a variety of specialized missions in Intelligence, Surveillance, and Reconnaissance (ISR). According to the Defense Advanced Research Projects Agency (DARPA) the MAV is a small aerial robot capable of exploiting its small size and mobility for a broad range of remote operations (14). The MAV must provide a stable, inexpensive, and expendable surveillance platform (21). Similarly to the gun platforms provided by larger aircraft, the MAV must provide precise directional control for mission success, and it must do this in a very demanding flight regime. Because they have small dimensions and must fly at low speeds, the Reynolds numbers experienced by typical MAV's range from 10^4 to 10^5. At such low Reynolds numbers the vehicle is susceptible to sudden separation and loss of lift due to very large gusts. On a typical day the wind may vary by 10 mph, which is a significant velocity component for an aircraft traveling at 10 to 30 mph (10).

MAVs are highly energy and weight constrained. Because of this, all components of the MAV must be more tightly integrated than those of a traditional aircraft, and exploiting multi-functionalities across components becomes crucial to achieving useful MAV performance (14). Of particular interest along these lines is the impact on performance of a highly flexible wing at low Reynolds number.

The flexible wing is desirable for a variety of reasons. Logistically, the flexible-wing has a smaller packaging footprint which is crucial for its success as a man-portable platform. Aerodynamically, it passively adapts to local flow disturbances, providing more constant lift in gusty conditions. This process is termed "adaptive washout", and acts by the gust producing a deformation in the wing which reduces its lifting efficiency, but the higher dynamic pressure in the gust leads to near constant lift production (10). The dynamic response of the wing structure must be *tuned* to provide the correct amount of deformation with gust onset, while at the same time meeting packaging and strength requirements.

The already recognized importance of fluid-structure interaction in large aircraft design becomes a dominant consideration for the MAV, which is especially sensitive because of its light weight and large surface to volume ratio (14). The flexible wing thus provides a challenging opportunity to exploit aerodynamic, logistic and control benefits in the MAV design.

1.1 Research Objectives

The goal of this research is to implement a closely-coupled aeroelastic model using a combination of computational and experimental tools. The model will be evaluated against a wind tunnel characterization of a rigid and flexible wing MAV. The structural model will rely on experimentally determined mode shapes which are coupled with a time-accurate CFD solution. The model addressed in this paper allows the unsteady behavior of the MAV to be evaluated, and the quasi-static deformation's effect on performance can be compared to that measured in the wind-tunnel testing.

1.2 Prior Experimental Work

The MAV configuration investigated in this thesis was evaluated in previous wind-tunnel tests by Deluca (7). The flexible wing MAV, shown in Figure 1, has a 24 in wingspan, and is constructed of carbon fiber and parachute nylon. The wing has a carbon fiber leading edge and chord-wise carbon fiber ribs spanned by the parachute nylon, creating some resemblance to a bat wing. The rigid wing MAV shown in Figure 1 is similar to the flexible wing version except that the wing is constructed entirely of carbon fiber (further physical details in Table 1).

The results presented in (7) are time-averaged force measurements, and establish the static stability of the aircraft. The wind tunnel tests show that the static deformation of the flexible wing at high angle of attack increases the maximum lift to drag ratio, as well as maximum lift coefficient over the rigid wing. Also, as noted by DeLuca (15), the unsteady behavior of a possible laminar separation bubble just aft of the peak suction has a significant affect on the MAV's performance and is very sensitive to flow disturbances or small vibrations of the wing.

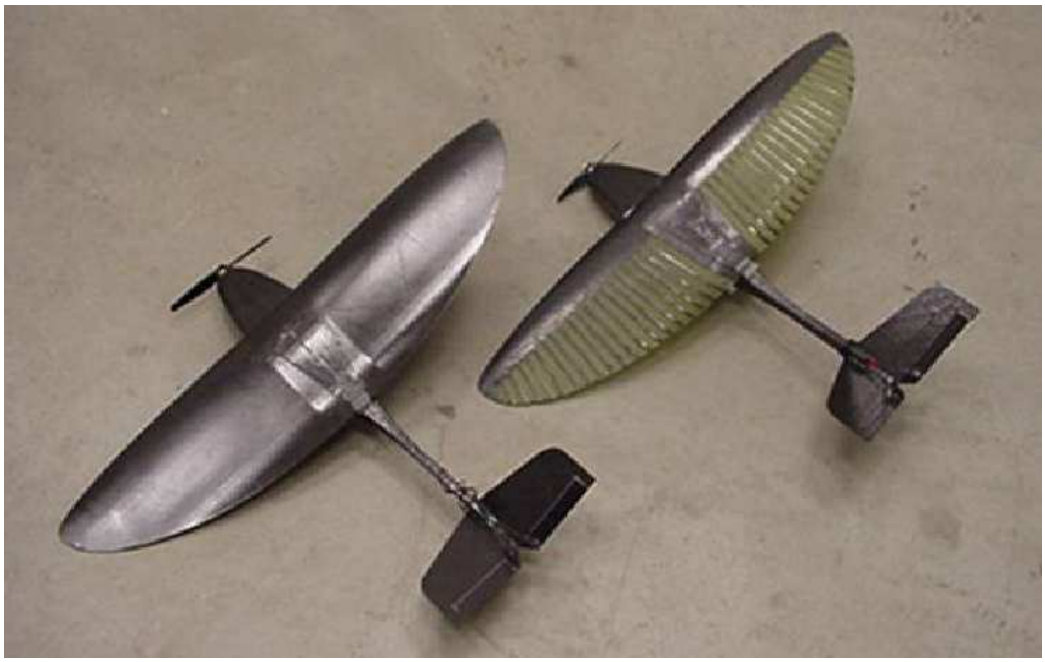

Figure 1: Flexible and rigid wing MAVs

Table 1: MAV physical properties

	Flexible wing	Rigid Wing	
Mass	0.320	0.360	kg
Plan-form Area	6.032×10^{-2}	6.032×10^{-2}	m^2
Root Chord	0.1524	0.1524	m
Mean Aerodynamic Chord	0.1067	0.1067	m
Span	0.6096	0.6096	m
Leading edge thickness	6.350×10^{-4}	6.350×10^{-4}	m
Membrane thickness	1.270×10^{-4}	1.270×10^{-4}	m
Aspect Ratio	6.1604	6.1604	

Other researchers have conducted wind tunnel testing of MAVs with membrane wings which have shown similar improvements in performance with the wind tunnel results presented in (7). A slightly smaller series of vehicles examined in (21) show the benefit of membrane over rigid wings in stall characteristics. Another interesting aspect of membrane wing performance noted in (21) is the adaptive washout of the flexible wing during gust and at high angles of attack. This provides improved performance by smoothing the production of lift, while at the same time decreasing the weight of the aircraft over that of the rigid wing.

1.3 Computational Modeling of Membrane Wings

A series of slightly smaller rigid and flexible wing MAV's are investigated in (11) using a closely-coupled method incorporating a Pressure Implicit with Splitting Operators (PISO) algorithm to solve the incompressible Navier-Stokes equations and a hyperelastic Moody-Rivlin model, based on the approach presented in (12). They have shown that the billowing of the membrane can have an offsetting affect on lift production to the adaptive washout, so the lift might be little changed from the rigid wing. Their results also show high frequency membrane vibration in steady free-stream (11). The membrane deformations at nominal to high angle of attack tend to increase the maximum lift-to-drag ratio.

1.4 Research Approach

The implementation and validation of the aeroelastic model begins with steady, non-deforming solutions to establish a base-line comparison with the rigid wing wind-tunnel results. Static structural deformations are compared to the flexible wing wind-tunnel measurements to evaluate the validity of the linear modal model. The flexible wing's effect on dynamic performance is measured by a simple gust response.

Particular attention is paid to the contributions to lift variation of the flexible wing under steady flight conditions as well as gust since lift production of the MAV is a critical factor in its performance and mission accomplishment.

1.5 Thesis Organization

Chapter I introduces the MAV concept, prior work in the field, and the current research approach. Chapter II presents a detailed description of the solution methods and the organization of the overall solution algorithm. The results are presented in Chapter III, with conclusions following in Chapter IV.

II. Research Methods

In general the computational aeroelastic problem is a three field problem, with conservation equations governing the fluid and structure, and some arbitrary choice of governing equations for the mesh geometry in each zone of the simulation (12). There are several approaches to this type of problem. In a *closely-coupled* solution the surface forces are transfered between the fluid solution and the structural model at each time step or iteration, while a *loosely-coupled* solution performs updates of deformations at a less frequent interval (13). In a *fully-coupled* solution the equations of motion for the fluid, structure, and grid are reformulated into one set of governing equations (9). Figure 2 shows the organization of the high level solver components in a computational aeroelastic analysis. The current method is *closely-coupled*; it communicates loads between the fluid and structure and updates the grid at each time step.

No matter the solution approach, communication between the computational domains of the fluid and structural simulations is a limiting factor on the accuracy of the aeroelastic analysis. The problem of communication between the problem domains can be eliminated by making the fluid and structure meshes point-matching along the surface interface. This requires the preliminary analysis of the problems to be very closely coordinated and forces somewhat unreasonable constraints on the two grids. The computational meshes required to accurately and efficiently solve the two problems differ greatly in complexity. CFD grids are usually stationary, and the equations of fluid dynamics are posed in an Eulerian frame. The CFD meshes are also highly stretched near solid surfaces to capture gradients due to viscosity. The structural equations of motion are usually posed in a Lagrangian reference

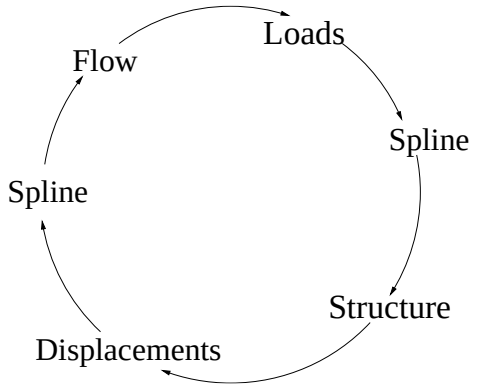

Figure 2: General aeroelastic algorithm

frame, and the grid deforms with the structure. This mesh tends to be much more isotropic than the CFD mesh, and the resolution requirements for acceptable accuracy are much less stringent. This is especially true in the case of a modal analysis, where a very simplified model of a complex aircraft geometry is often sufficient to accurately represent the lower, energy containing, modes of dynamic motion (1).

The structure's time response can be calculated by directly integrating the full equations of motion, or integrating the equations on a reduced basis of the lower modes. The modal approach has the advantage of requiring only one solution of the structural problem, and the flexibility of using either computationally or experimentally determined mode shapes and frequencies. Alternatively, the direct integration approach requires the full structural simulation to be solved concurrently with the fluid dynamics calculations. This added cost is hardly justified for problems dominated by "slow" physics, that is, the dominant frequencies of the structural response are much lower than the characteristic frequencies of the flow. Thus a modal analysis takes advantage of two efficiencies. The mesh size is reduced, and only one solution or experiment for the structure portion of the aeroelastic problem is needed.

2.1 Fluid Solution: Navier-Stokes

The Navier-Stokes equations are a continuum formulation for the conservation of mass, momentum, and energy in a fluid. The integral form of the governing equations is

$$\frac{\partial}{\partial t} \iiint_V \mathbf{W} dV + \oiint_A (\mathcal{F} - \mathcal{F}_v) \cdot d\vec{A} = 0 \tag{1}$$

Where the vector of conserved variables, and the fluxes are

$$\mathbf{W} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix} \qquad \mathcal{F} = \begin{bmatrix} \rho U \\ \rho u U + p n_x \\ \rho v U + p n_y \\ \rho w U + p n_z \\ (\rho E + p) U \end{bmatrix} \qquad \mathcal{F}_v = \begin{bmatrix} 0 \\ n_x \tau_{xx} + n_y \tau_{xy} + n_z \tau_{xz} \\ n_x \tau_{yx} + n_y \tau_{yy} + n_z \tau_{yz} \\ n_x \tau_{zx} + n_y \tau_{zy} + n_z \tau_{zz} \\ n_x \Theta_x + n_y \Theta_y + n_z \Theta_z \end{bmatrix} \tag{2}$$

The contravariant velocity, U, is

$$U = \vec{v} \cdot \vec{n} = un_x + vn_y + wn_z \tag{3}$$

Θ in Equation 2 is given by

$$\Theta = \begin{bmatrix} u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\partial T/\partial x \\ u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\partial T/\partial y \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\partial T/\partial x \end{bmatrix} \tag{4}$$

The shear stress tensor (following Stoke's hypothesis) is given by

$$\tau_{xx} = \frac{2}{3}\mu\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial w}{\partial z}\right) \tag{5}$$

$$\tau_{xy} = \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \tag{6}$$

$$\tau_{yy} = \frac{2}{3}\mu\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x} - \frac{\partial w}{\partial z}\right) \tag{7}$$

$$\tau_{xz} = \tau_{zx} = \mu\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right) \tag{8}$$

$$\tau_{zz} = \frac{2}{3}\mu\left(2\frac{\partial w}{\partial z} - \frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}\right) \tag{9}$$

$$\tau_{yz} = \tau_{zy} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \tag{10}$$

An assumption of incompressibility (constant density) leads to a simplified form of the governing equations, which decouples the conservation of energy from the mass and momentum equations.One method of solving this set is elaborated in section 2.1.1.

2.1.1 Segregated Incompressible. The low-velocity, segregated, semi-implicit method for pressure-linked equations (SIMPLE) method is an iterative procedure to solve the incompressible Navier-Stokes equations. It applies a pressure correction to the decoupled, incompressible continuity and momentum relations (19). The pressure is written as

$$p = p_0 + \acute{p} \tag{11}$$

where p_0 is the guessed value of pressure and $\acute{p}$ is a pressure correction. Similarly, the velocity components are written as

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} u_0 \\ v_0 \\ w_0 \end{bmatrix} + \begin{bmatrix} \acute{u} \\ \acute{v} \\ \acute{w} \end{bmatrix} \tag{12}$$

The velocity corrections are computed using an approximate form of the momentum equation

$$\begin{aligned} \rho\frac{\partial \acute{u}}{\partial t} &= -\frac{\partial \acute{p}}{\partial x} \\ \rho\frac{\partial \acute{v}}{\partial t} &= -\frac{\partial \acute{p}}{\partial y} \\ \rho\frac{\partial \acute{w}}{\partial t} &= -\frac{\partial \acute{p}}{\partial z} \end{aligned} \tag{13}$$

The velocity correction is assumed to be zero at the previous iteration step so the left hand side of Equation 13 can be simplified as

$$\rho\frac{\acute{u}^{n+1} - \acute{u}^{n}}{\Delta t} = \rho\frac{u^{n+1}}{\Delta t} \tag{14}$$

Substituting Equation 14 into 13 gives the current velocity corrections as

$$\begin{aligned} \acute{u} &= -A\frac{\partial \acute{p}}{\partial x} \\ \acute{v} &= -A\frac{\partial \acute{p}}{\partial y} \\ \acute{w} &= -A\frac{\partial \acute{p}}{\partial z} \end{aligned} \tag{15}$$

where A is the pseudo-time step divided by the density. Substituting 12 and 13 into the continuity equation gives

$$\left(\frac{\partial u_0}{\partial x} + \frac{\partial v_0}{\partial y} + \frac{\partial w_0}{\partial z}\right) + A\left(\frac{\partial^2 \acute{p}}{\partial x^2} + \frac{\partial^2 \acute{p}}{\partial y^2} + \frac{\partial^2 \acute{p}}{\partial z^2}\right) = 0 \tag{16}$$

which is solved for the pressure corrections, which are then used to update the pressure and velocity field. A central difference is generally used in equation 13 for the $\acute{p}$ derivatives, which can result in odd-even decoupling, for this reason the SIMPLE method is generally solved on a staggered grid to prevent odd-even decoupling in the pressure field (19). We use the commercial CFD package, FLUENT, which implements the above algorithm. The

velocity gradients are found using either a first or second order upwind discretization (in FLUENT). Both of the schemes are used in the present research. The effect of the spatial order of accuracy of the velocity gradient on the solution is examined.

2.2 Force Transformation: Infinite Plate Spline

The structure and fluid grids are linked by the "infinite plate spline" approach implemented by Parker (16) for use with FLUENT. This implementation has already shown accurate results for several transonic flutter cases (16). The spline determines the positions of the CFD grid points, q_a, by minimizing the bending energy of a thin plate passing through the structural grid points, q_s (8). The CFD grid points are given by

$$\{q_a\} = [S]\{q_s\} \tag{17}$$

where the spline matrix, S, is

$$S = B_{as}\left[B_s^{-1} - B_s^{-1}R_s\left(R_s^T B_s^{-1} R_s\right)^{-1} R_s^T B_s^{-1}\right]^{-1} + R_a\left(R_s^T B_s^{-1} R_s\right)^{-1} R_s^T B_s^{-1} \tag{18}$$

with R and B given by

$$R = \begin{bmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & y_n & z_n \end{bmatrix} \tag{19}$$

$$\begin{aligned} B_{as} &= r_{as}^2 \ln(r_{as}^2) \\ r_{as}^2 &= (x_{ai} - x_{sj})^2 + (y_{ai} - y_{sj})^2 + (z_{ai} - z_{sj})^2 \\ B_s &= r_s^2 \ln(r_s^2) \\ r_s^2 &= (x_{si} - x_{sj})^2 + (y_{si} - y_{sj})^2 + (z_{si} - z_{sj})^2 \end{aligned} \tag{20}$$

The subscript a in Equations 18, 19, and 20 denotes a matrix of coordinates from the CFD grid, the subscript s denotes a matrix of coordinates from the structure grid, and the subscript as denotes a matrix containing both coordinates (16). The forces can be

transformed to the structure grid simply by

$$F_s = S^T F_a \tag{21}$$

One drawback of using FLUENT is that the forces are updated at every time-step, but not every sub-iteration of the implicit solver. This limitation is due to the interface available for user defined functions in FLUENT. This is unfortunate since the forces must be time lagged for use in the structural model. This has a detrimental effect on the temporal accuracy that can be achieved with an implicitly integrated scheme, or the commonly used multi-stage explicit schemes. In addition, the fluid solution is limited to first order temporal and spatial accuracy for solutions with deforming meshes in FLUENT.

2.3 Structural Solver: Modal Analysis

The harmonic response of a structure is described by

$$\left(K - \omega^2 M\right) d = 0 \tag{22}$$

where K is the stiffness matrix, M is the mass matrix, ω is a natural frequency, and d is the vector of displacements of the nodes. Harmonic vibration is a state of balance between the forces due to inertia ($\omega^2 M d$) and the elastic forces (Kd) (6). Rewriting equation 22 makes it easy to recognize the generalized eigen-problem

$$Kd = \omega^2 M d \tag{23}$$

with the eigenvalues being a natural frequency squared and the eigenvector being the vector of displacements, or modes, which satisfies equation 23 for the corresponding eigenvalue (1). This research uses experimentally determined eigenvalues and eigenvectors based on laser vibrometery measurements of the dynamic response of the MAV in air and in vacuum.

2.3.1 Structural equation of motion on a reduced basis. The advantage of modal analysis is that the model uses a few of the low frequency, energy containing modes as a reduced-dimension basis for the structural dynamics problem. The system can be solved

more efficiently since the orthogonal basis, Φ, allows an uncoupled solution to the structural system, given by

$$\Phi^T K \Phi d = \omega^2 \Phi^T M \Phi d \tag{24}$$

We first seek to diagonalize the stiffness matrix, K so that the structural model can be efficiently solved as a decoupled system. Noting that K is a real symmetric matrix, and recalling the *spectral theorem*, which states that a real symmetric matrix is diagonalizable by an orthogonal matrix with the eigenvectors of the matrix we wish to diagonalize as its columns(18), we have our orthogonal matrix, Φ which has the property:

$$\Phi^T \Phi = I \tag{25}$$

which gives us the inverse as well, $\Phi^{-1} = \Phi^T$. Φ, sometimes referred to as the *modal matrix*, has the eigenvectors of the stiffness matrix as its columns.

Generally, the mode shapes are mass normalized such that

$$d^T M d = 1 \tag{26}$$

Also, the eigenvectors are mass-orthogonal with each other (shown in Appendix A), that is

$$d_i^T M d_j = 0 \quad (i \neq j) \tag{27}$$

Usually the mass matrix is a known quantity that is developed as a preliminary step in a structural analysis, such as the finite element method. For our purposes we assume a uniform mass distribution, giving a mass matrix of the form

$$M = \left(\frac{m_{total}}{n}\right) I \tag{28}$$

where n is the number of nodes, and m_{total} is the total mass of the structure, which is a lumped-mass matrix assuming uniform mass distribution. The mass normalization of the

eigenvectors then is given by

$$\begin{aligned} d^T M d &= \frac{m_{total}}{n} c^2 \\ d_{norm} &= \left(\frac{\sqrt{n}}{c\sqrt{m_{total}}} \right) d \end{aligned} \tag{29}$$

In Equation 29 the value of the inner product before normalization is represented by c^2. The measured frequencies of the MAV's harmonic response give the mass to stiffness ratio, but the problem of determining the masses of the structure's grid points is under-constrained unless n modes are measured. This is unlikely, since the number of points needed for adequate spatial resolution would require prohibitively high frequency modes to be excited. This leaves the choice of weights a somewhat open question.

Unlike the eigenvectors from a mathematical model, the experimentally measured eigenvectors, or mode shapes, will not in general be orthogonal for a variety of reasons. The structure has some non-linearity which manifests itself as coupling between the mode shapes. In addition, experimentally exciting a single mode perfectly is difficult, so the measured shape will have some small components of the other modes present.

We use the Gram-Schmidt process to remove the non-orthogonal components from the modes (18). The lowest frequency mode is retained unaltered (except for normalization), and the parallel components of each subsequent mode are removed. The algorithm is straightforward and shown in Equations 30 and 31, with a full listing in Appendix B.

For all of the j vectors:

1. Subtract off the parallel components of previous vectors

$$q_j = a_j - \sum_{i=1}^{j-1} \left(q_i^T a_j \right) q_i \tag{30}$$

2. Normalize

$$q_j = \frac{q_j}{\|q_j\|} \tag{31}$$

The *modal matrix* is now

$$\Phi = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \tag{32}$$

which is used to diagonalize the structure's stiffness matrix

$$\left[\omega^2\right] = \Phi^T K \Phi \tag{33}$$

The *spectral matrix* in equation 34 is given by

$$\left[\omega^2\right] = \begin{bmatrix} \omega_1^2 & 0 & \cdots & \\ 0 & \omega_2^2 & 0 & \cdots \\ & \ddots & \ddots & \ddots \\ & \cdots & 0 & \omega_n^2 \end{bmatrix} \tag{34}$$

An arbitrary displacement vector d, or acceleration vector $\ddot{d}$ is written as a linear combination of the modes.

$$\begin{aligned} d &= \Phi z \\ \ddot{d} &= \Phi \ddot{z} \end{aligned} \tag{35}$$

where z is the vector of *modal displacements*. Substituting equation 35 into the undamped equation of motion gives

$$M\Phi\ddot{z} - K\Phi z = F_{ext} \tag{36}$$

Pre-multiplying equation 36 by Φ^T gives

$$\Phi^T M \Phi \ddot{z} - \Phi^T K \Phi z = \Phi^T F_{ext} \tag{37}$$

With the eigenvectors mass-normalized, substituting the spectral stiffness matrix from equation 33 gives the modal equation of motion (with no damping):

$$\ddot{z} + \left[\omega^2\right] z = \Phi^T F_{ext} \tag{38}$$

Truncating the mode set to the first m modes gives

$$d \approx \sum_{i=1}^{m} \Phi_i z_i$$

$$\ddot{d} \approx \sum_{i=1}^{m} \Phi_i \ddot{z}_i$$

where Φ is now an $m \times n$ matrix. Equation 38 is integrated in time using this reduced basis.

Two separate formulations for the time integration of the governing equations of the structure are examined, an explicit multi-stage scheme, and a semi-implicit method.

2.3.2 Multi-stage Explicit. We use a four stage CFD style Runge-Kutte scheme (source listing for all integration routines in Appendix C) to integrate the structural equations of motion. The stage coefficients are

$$\alpha_1 = \frac{1}{4}, \alpha_2 = \frac{1}{3}, \alpha_3 = \frac{1}{2}, \alpha_4 = 1 \tag{39}$$

The second order system given by equation 38 can be written as a coupled system of first order ODE's.

$$\frac{\partial}{\partial t} \begin{bmatrix} z \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \dot{z} \\ \Phi^T F - \omega^2 z \end{bmatrix} \tag{40}$$

which can be integrated by an arbitrary multistage scheme by

$$\begin{bmatrix} z \\ \dot{z} \end{bmatrix}^{k+1} = \begin{bmatrix} z \\ \dot{z} \end{bmatrix}^{n} - \alpha^k (\Delta t) \begin{bmatrix} \dot{z} \\ \Phi^T F - \omega^2 z \end{bmatrix}^{k} \tag{41}$$

where k denotes the stage level and n denotes the time level, the final stage gives the $n+1$ solution. For an m stage scheme at least second order accuracy is achieved with $\alpha^{m-1} = 0.5$ and $\alpha^m = 1$ (2).

2.3.3 Newmark – semi-implicit. The Newmark method makes substitutions for displacements and velocities at the $n+1$ time level with Taylor series expansions about the n time level allowing the acceleration to be updated using only n time level values. Then a new velocity can be calculated based on an average of the acceleration between the n and $n+1$ time levels (16).

$$\ddot{z}^{n+1} = \frac{F_z^n - \omega^2 \left(z^n + (\Delta t)\dot{z}^n + (\Delta t)^2 \ddot{z}^n / 3 \right)}{1 + (\Delta t)^2 \omega^2 / 6} \tag{42}$$

$$\dot{z}^{n+1} = \dot{z}^n + \frac{\Delta t}{2} \left(\ddot{z}^n + \ddot{z}^{n+1} \right) \tag{43}$$

$$z^{n+1} = z^n + (\Delta t)\dot{z}^n + \frac{(\Delta t)^2}{6}\left(2\ddot{z}^n + \ddot{z}^{n+1}\right) \tag{44}$$

The performance of both of these schemes is evaluated in section 3.4 by integrating the free-vibration of the modes in response to a displacement perturbation, and then they are applied to the full aeroelastic problem.

2.4 Laser Vibrometry

Laser vibrometry consists of exciting the structure and then measuring the phase shift in a laser beam reflected off of the vibrating structure against a reference beam split from the same source(17). The unit used for this research allows three measurement heads to be positioned to determine velocity and displacements of a vibrating structure in three dimensions. However, this analysis assumes that the main deformations are in the direction normal to the chord line of the wing so only one measurement head is needed.

The experimentally measured frequency is in units of cycles per second (Hz). The ω in equations 22 and 23 is a circular frequency (radians per second). The simple, but important conversion is

$$\omega^2 = (2\pi f)^2 \tag{45}$$

The air around the vibrating structure adds an "apparent mass" effect to the measured frequencies (5). Because the flexible wing is so light the effect of the apparent mass of air is significant so this effect needs to be removed from the measured mode frequencies so they can be used in the aeroelastic solution. We achieve this by measuring the vibrations of the wing in a vacuum chamber at varying ambient pressure levels. This data is used to develop a frequency correction to the modes used in the model.

The vacuum chamber has limited volume and optical access, so only the wing of the MAV, separate from the fuselage, was used in the variable ambient pressure tests. This is a reasonable way to develop the apparent mass correction since the main affect will be on the relatively low inertia (mass) membrane wing rather than the significantly more massive

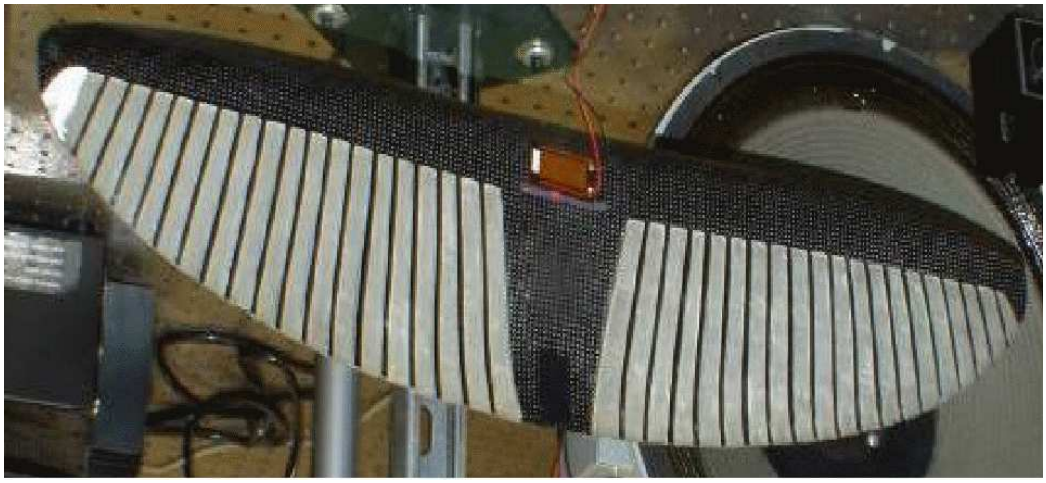

Figure 3: Membrane wing on shaker table

fuselage. Changes to the mode shapes due to the wing being removed from the fuselage are not critical since the correction is made to the frequency of the mode, not its shape.

2.4.1 Wing. Measurements of the wing's motion are made by the scanning head at specific locations over the wing's surface. These locations become the "grid" for the structural model. The damped mode shapes are measured easily in air by exciting the model with a large speaker. Figure 3 shows the wing attached to the shaker table with the speaker used to excite the structure in the background. These mode shapes are shown in Figure 4. Figure 4 shows the very noisy mode shapes at the higher frequencies. This is likely caused by removing the wing from the fuselage which removes the constraint on the central portion of the wing's trailing edge where it attaches to the rest of the aircraft.

The undamped modes are measured in a vacuum chamber with a small port for optical access, and the model is excited by a small piezo-electric patch attached to the wing. Due to the restricted optical access in the vacuum chamber, the mode frequencies are measured by a single point scan on the wing, chosen in a location that is likely to have a large amplitude response for all of the modes. This allows frequency information to be measured, but no shape information can be obtained from a single measurement point. Therefore the mode shapes themselves are measured in air with the multi-point scan over the entire wing.

The correction to the frequency of the damped shape (measured in air) is made based on the measured frequency variation of the vacuum results (3).

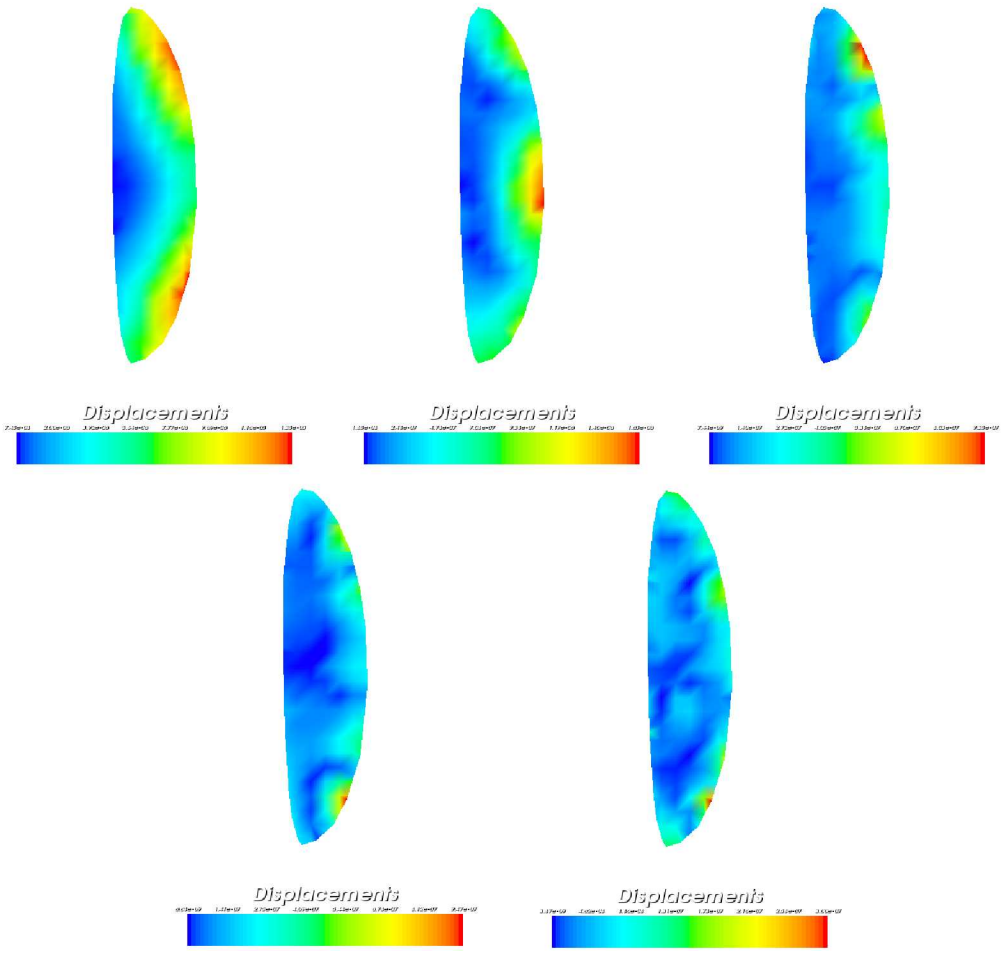

Figure 4: Wing only measured mode shapes

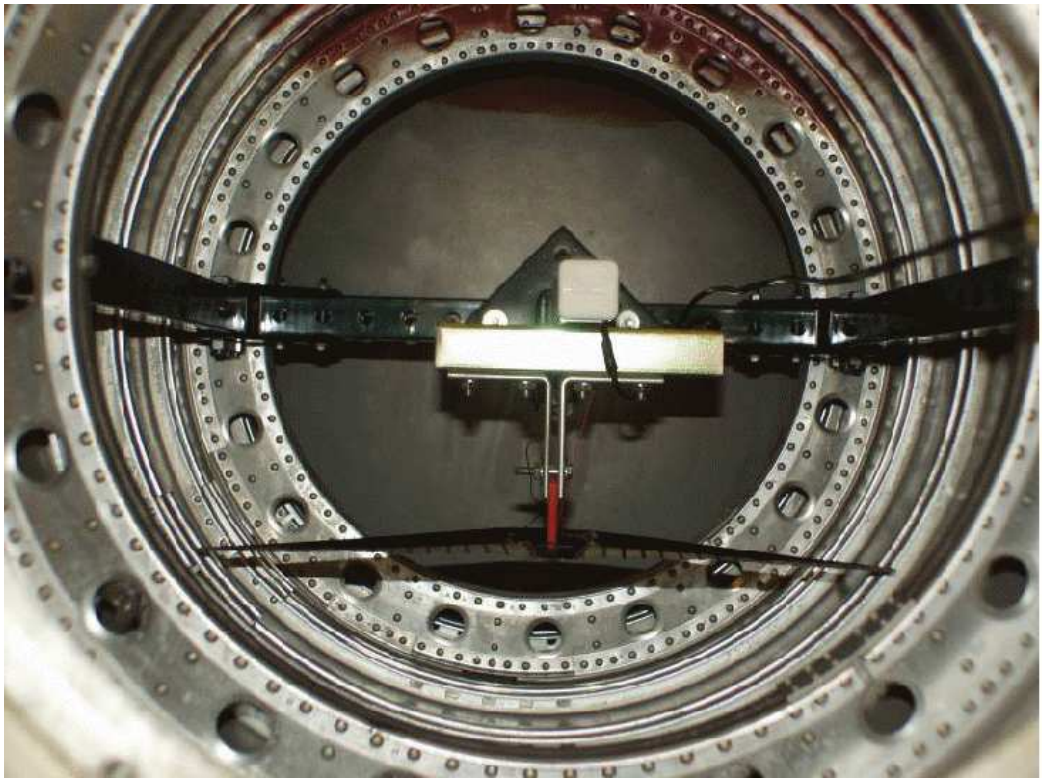

Figure 5: Membrane wing in vacuum chamber

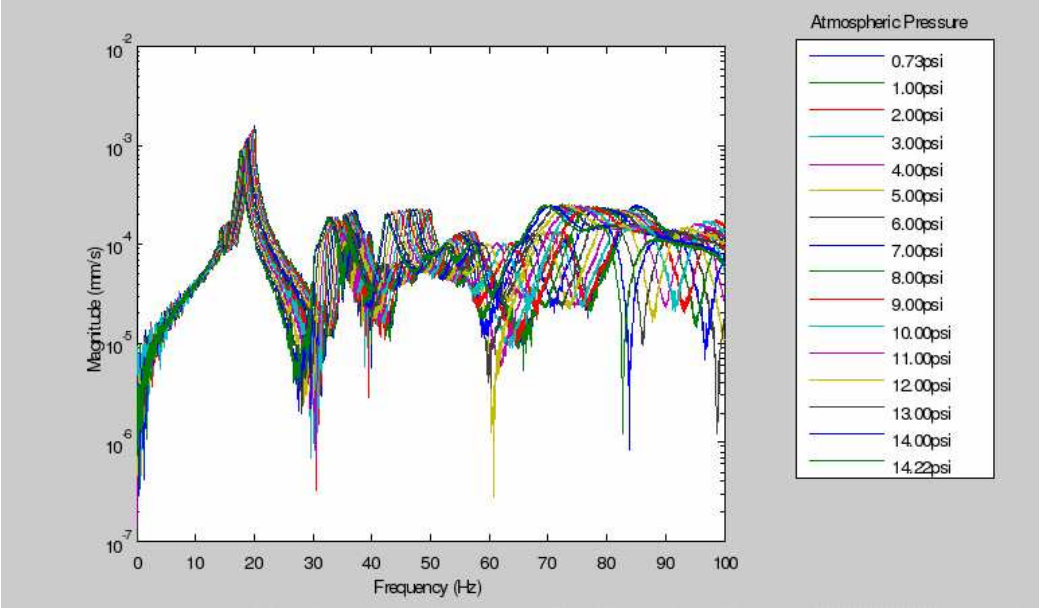

Figure 6: Effect of ambient pressure on mode frequencies

The apparent mass of the air moving around the vibrating structure causes the MAV's wing to vibrate at a lower frequency in air (5). The shift is most easily seen on the power spectrum in Figure 6 for the second mode. The spike around 20 Hz shifts to a larger amplitude and higher frequency as vacuum is approached. We use a simple linear extrapolation (equation 46) to remove the effect of air on the wing's vibration. Note: the validity of this adjustment will be evaluated in a separate research effort.

$$f_{adjust} = f_{measured}\,(1 + s) \tag{46}$$

where s is based on the frequency shift of the second mode shape, shown in detail in Figure 7. The legend of Figure 7 also shows the decrease in measured damping (2% to 1.3%) as the air is removed from the chamber.

$$s = \frac{19.9 - 17.6}{19.9}$$

The higher modes are not excited very reliably by the peizo-electric patch for the variable ambient pressure tests. Figure 6 shows the lack of well defined amplitude peaks for any of the higher modes.

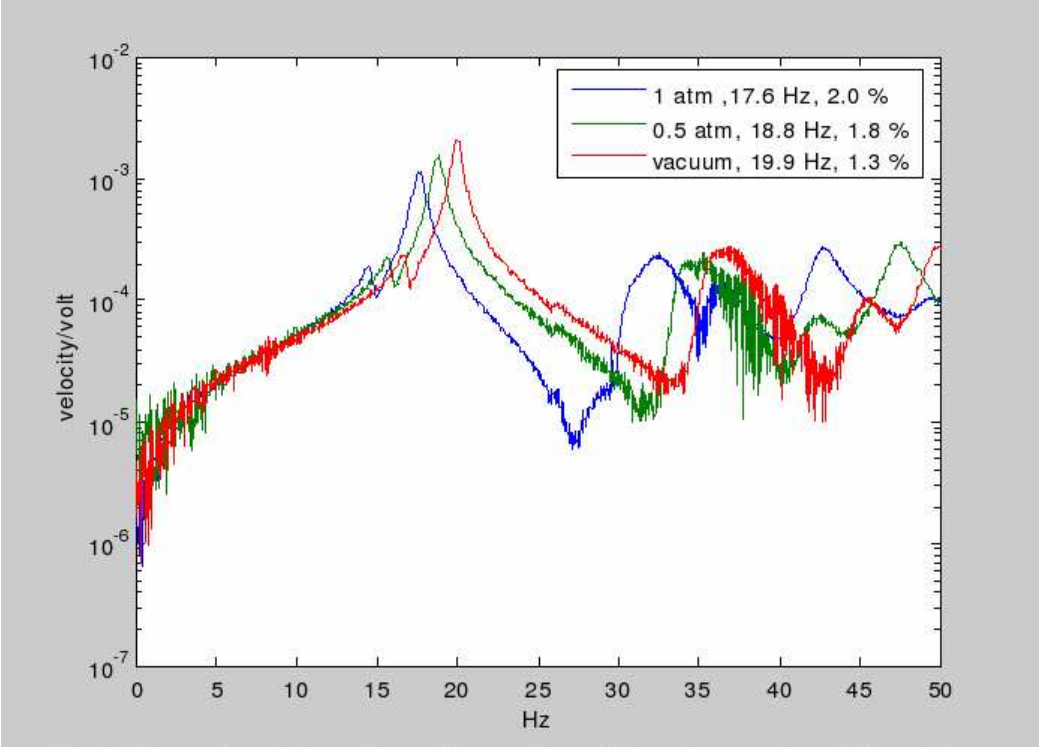

Figure 7: Frequency shift of 2nd mode due to apparent mass

2.4.2 Full Aircraft. The full aircraft is shown mounted on the shaker table in Figure 8. The speaker used to drive the structure during the measurements is seen in the foreground of this picture, the scanning head is not visible in this view.

The measured mode shapes are shown in Figure 9. The magnitude of the displacements measured by the laser vibrometry scanning head are very small, but the magnitude of the mode vector is arbitrary, and will be normalized as part of the Gram-Schmidt process.

Table 2 shows the inner product of each eigenvector with every other, shown in equation 47.

$$\frac{d_i \cdot d_j}{\|d_i\|} \tag{47}$$

The non-zero off-diagonal components in Table 2 show that the raw measurements contain elements of the other mode shapes. This is understandable since we are exciting a real structure with an imperfect driver (the speaker), perfectly exciting just a single mode is unlikely. The closer the frequencies of the modes are to each other the more likely they will be coupled, or co-excited by the speaker. Notably the fourth and fifth modes are very close

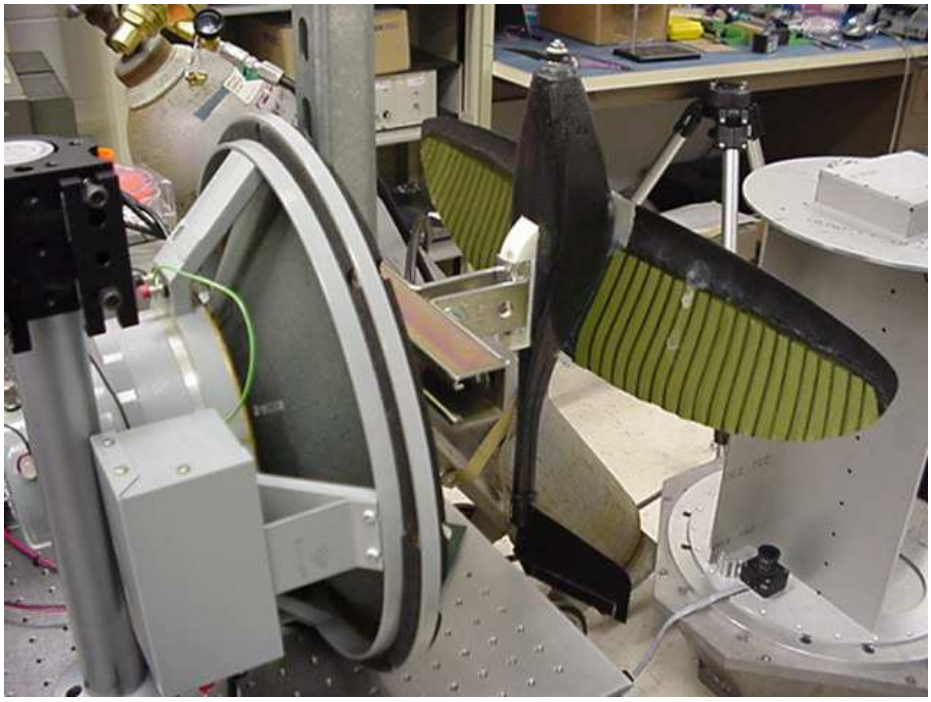

Figure 8: Full aircraft mounted on shaker table

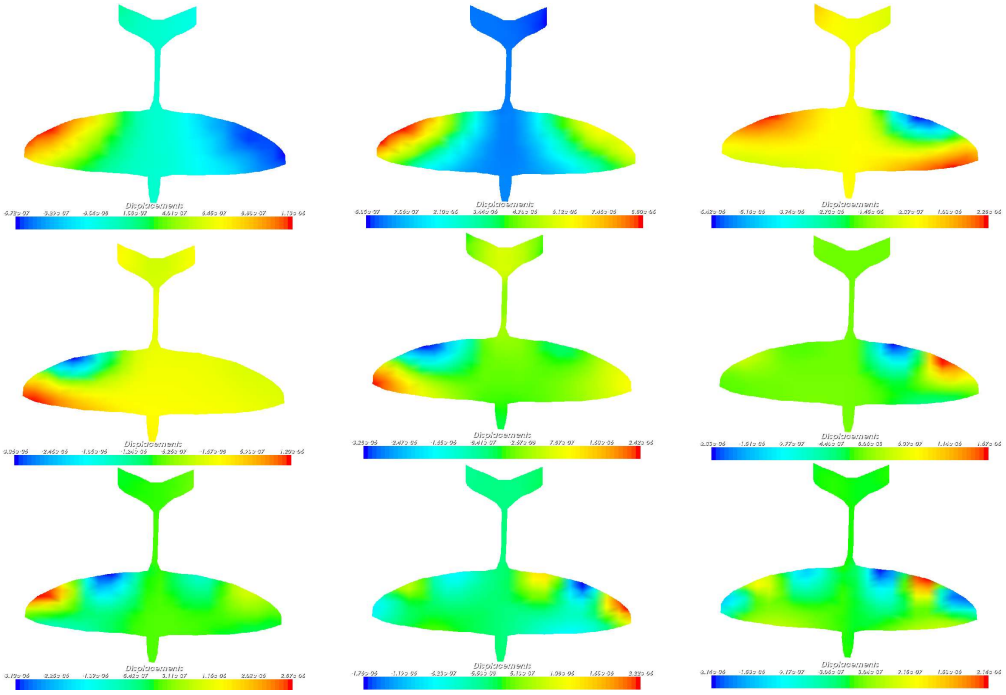

Figure 9: Full aircraft measured mode shapes

Table 2: Inner products of the nine measured mode shapes

1.00	0.37	0.40	-0.32	-0.19	-0.03	0.04	-0.11	-0.02
0.37	1.00	-0.11	-0.37	-0.11	0.13	0.08	0.06	-0.14
0.40	-0.11	1.00	-0.26	-0.10	0.07	0.00	0.09	-0.06
-0.32	-0.37	-0.26	1.00	0.83	-0.09	0.14	-0.05	-0.03
-0.19	-0.11	-0.10	0.83	1.00	0.12	0.33	0.07	-0.12
-0.03	0.13	0.07	-0.09	0.12	1.00	0.35	-0.12	0.16
0.04	0.08	0.00	0.14	0.33	0.35	1.00	0.07	0.20
-0.11	0.06	0.09	-0.05	0.07	-0.12	0.07	1.00	-0.52
-0.02	-0.14	-0.06	-0.03	-0.12	0.16	0.20	-0.52	1.00

together in frequency, and their components are very nearly parallel (inner product close to unity). The non-orthogonal components of the measured shapes must be eliminated so they can be used to diagonalize the stiffness matrix.

All of the off-diagonal components shown in Table 2 are around machine zero for the modes after the Gram-Schmidt process. The measured mode shapes are shown in Figure 9, and the result of applying the Gram-Schmidt process to achieve an orthonormal basis are shown in Figure 10.

In the ortho-normalization process nothing is subtracted off of the first mode, it is only normalized. The fourth and fifth mode are very close together in frequency and so there is significant overlap between the two measurements, referring back to Table 2, we see that the fourth and fifth mode are nearly coincident. The wing/fuselage deformations are roughly the same, but the tail orientation is slightly different, this results in a orthonormalized shape that is significantly different from the measured shape.

The power spectrum of the harmonic response of the MAV is shown in Figure 11. The frequency peak of the second mode shape is highlighted in the lower portion of the figure and this shape is illustrated in the upper portion.

The higher mode shapes of the detached wing bear little resemblance to the movement of the wing on the full aircraft. This is expected since the trailing edge is completely unrestrained. Referring back to Figures 6 and 7, we see that the higher modes are difficult to excite, and do not have very well defined frequency peaks. This is the reason for using

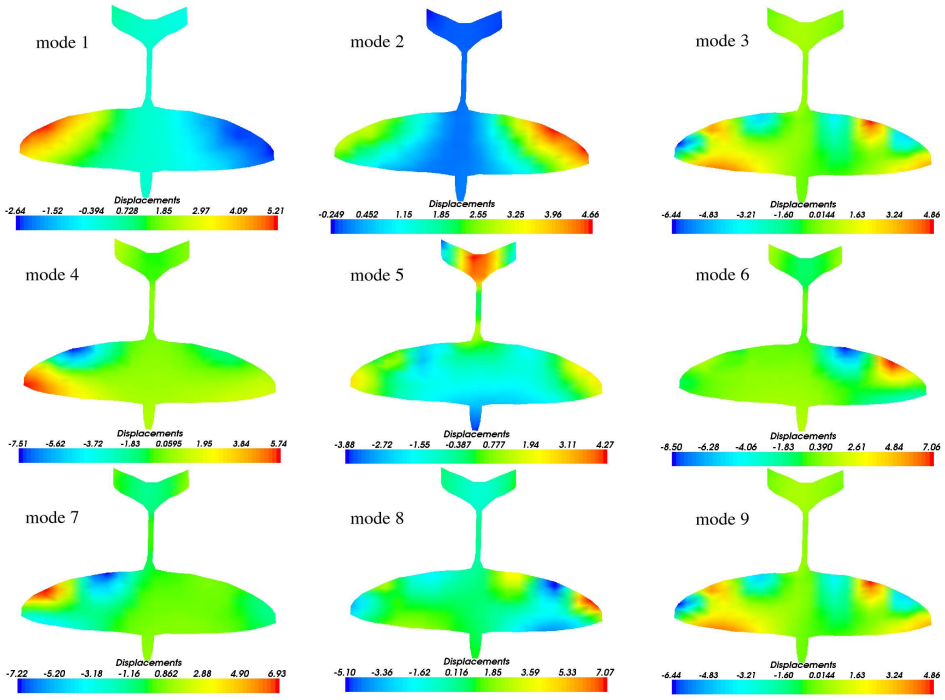

Figure 10: Orthonormalized eigenvectors

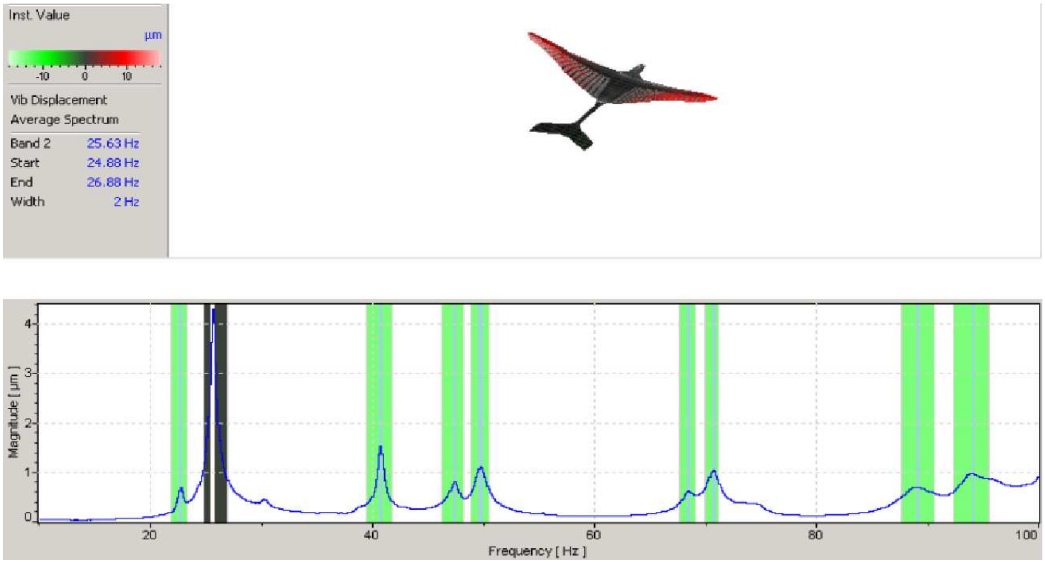

Figure 11: Power spectrum of harmonic response, and second mode shape visualization

Table 3: Full aircraft frequency adjustment

Mode	$f_{measured}$ (Hz)	f_{adjust} (Hz)
1	22.75	25.38
2	25.63	28.59
3	40.75	45.46
4	47.38	52.86
5	49.75	55.50
6	68.50	76.42
7	70.75	78.93
8	89.13	99.43
9	94.13	105.01

the more well defined frequency shift of the second mode to make the adjustments due to ambient pressure.

Table 3 shows the adjusted frequencies for the full aircraft. This frequency adjustment uses the measurements made on the wing only, but is applied to the frequencies measured for the whole aircraft configuration. This is reasonable since the greatest affect of the apparent mass is felt by the motion of the light, flexible wing, and the symmetric bending mode is similar for both configurations.

2.5 Solution Algorithm

The complete solution is accomplished following the procedure illustrated in Figure 12. The basic aeroelastic algorithm consists of a series of transformations. Forces calculated by FLUENT are transformed to the modal model, and displacements calculated by integrating the modes are transformed back to the fluid grid.

All of the transformation matrices, the infinite plate splines, and the modal basis, are calculated and stored once at the beginning of the solution. The proper multiplications are then made at each time step by calling user defined functions in FLUENT. In this way the structure gets forces from the fluid through the infinite plate spline at every time-step, and the fluid grid is moved according to the structural displacements, also transformed by the infinite plate spline at each time-step. Within the structural model, the forces are transformed from grid points to modes, which are integrated, and then transformed back to grid point displacements.

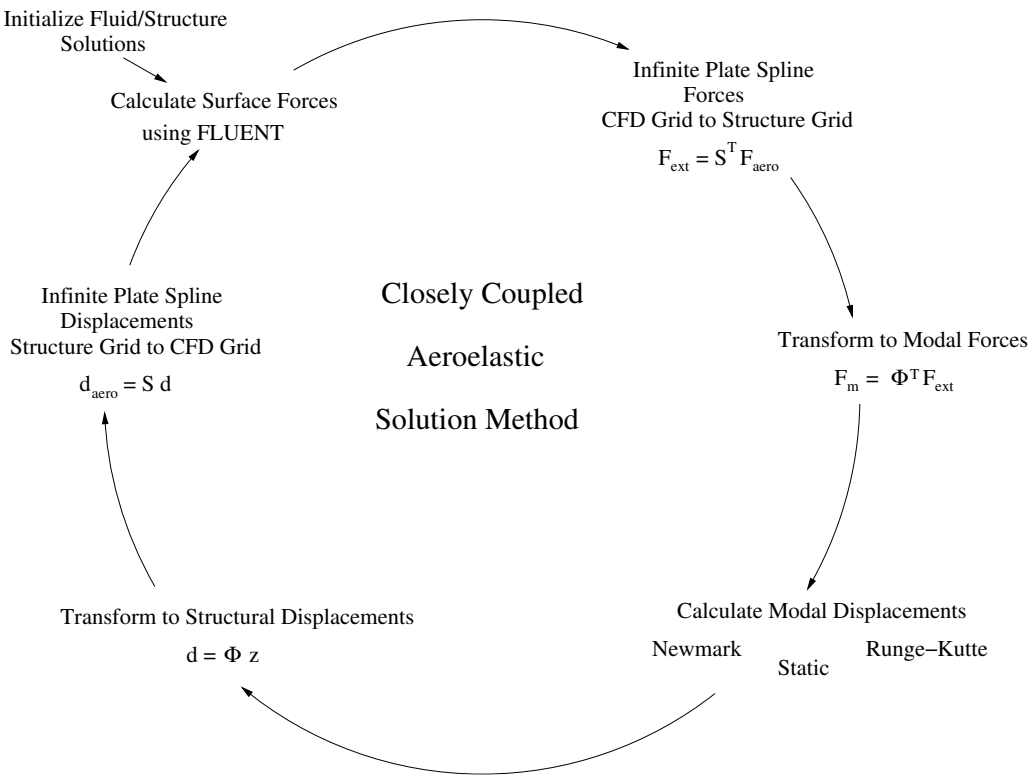

Figure 12: Solution algorithm flow chart

III. Results

The validation of the aeroelastic model begins by comparing the computational solutions with no deformations to the wind-tunnel results for the rigid wing MAV. The static deformations of the MAV are then incorporated into the steady-state fluid solution, and these results are compared to the time-averaged force measurements made in the wind tunnel on the flexible wing MAV. Finally, the dynamic response of the MAV in steady free-stream and gusty conditions is investigated.

3.1 Problem domain and computational grid

The MAV is modeled inside the wind tunnel which was used for the experimental investigation. This reduces the overall size of the grid required to accurately resolve the flowfield, but requires the volume grid to be re-triangulated for each different angle of attack. Figure 13 shows the MAV at 0° α in the wind tunnel test-section used for this research. The test section dimensions are 31"×44"×72", which gives a span-to-tunnel width ratio of 0.545 (7). The comparison to experimental results is also more meaningful since the wall effects are not removed in the computational solution as they would be if the MAV was modeled in free flight. The blockage correction applied in (7) amounts to less than 0.1% of the dynamic pressure, while the cross tunnel variation in velocity was measured to be on the order of 2% of the free-stream velocity, so this correction is neglected in the present calculations.

The fluid grid has approximately 640,000 tetrahedral cells, based on the geometry definition used for preliminary CFD investigation presented in (15). The cell volumes range from approximately 1.8×10^{-12} to 4.1×10^{-4} m^3. Figure 14 shows the surface grid of the MAV, and figure 15 shows the faces of the interior cells in a slice across the front portion of the MAV.

One drawback to using an unstructured grid for these viscous solutions is the large number of points required on the MAV surface. The unstructured grid spacing along the surface must be on the order of the boundary layer height to properly resolve this near wall feature with unit aspect ratio pyramids. A structured grid, on the other hand, can be highly skewed near the wall to achieve much a higher resolution in the surface normal direction compared to the surface tangent direction, reducing the number of nodes on the surface.

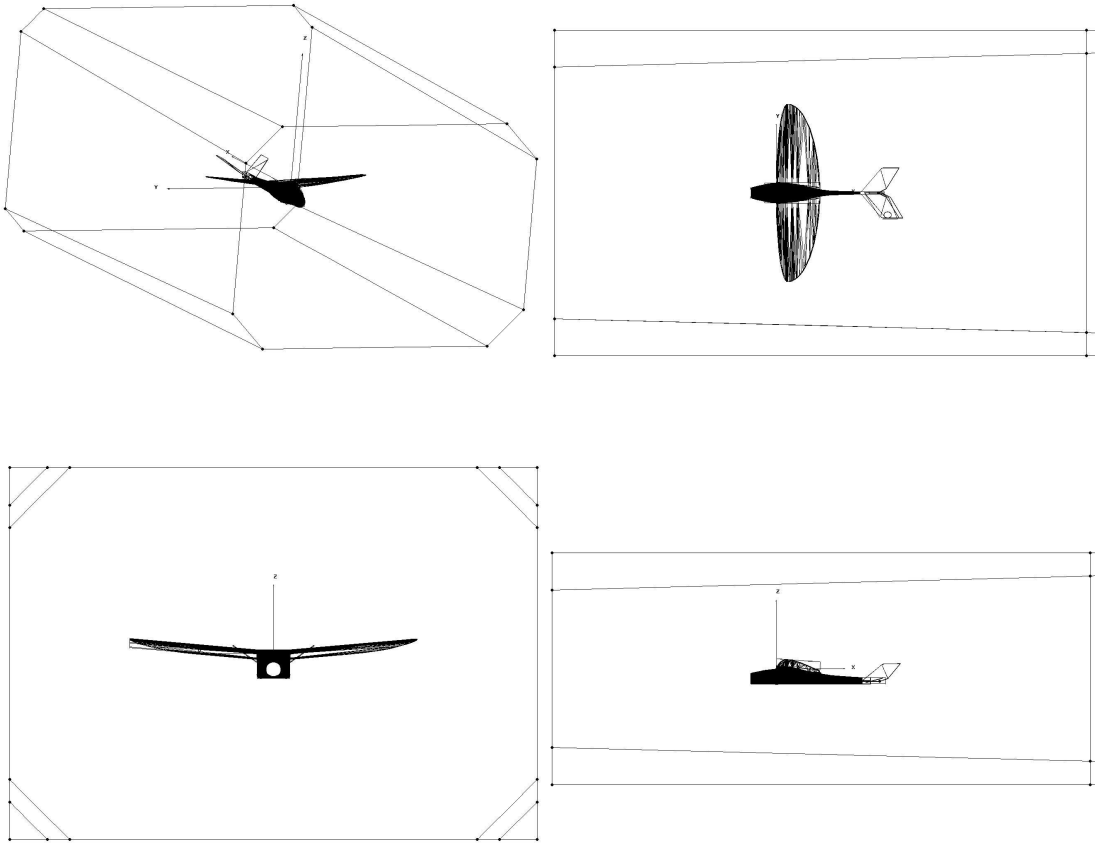

Figure 13: MAV in wind tunnel test-section

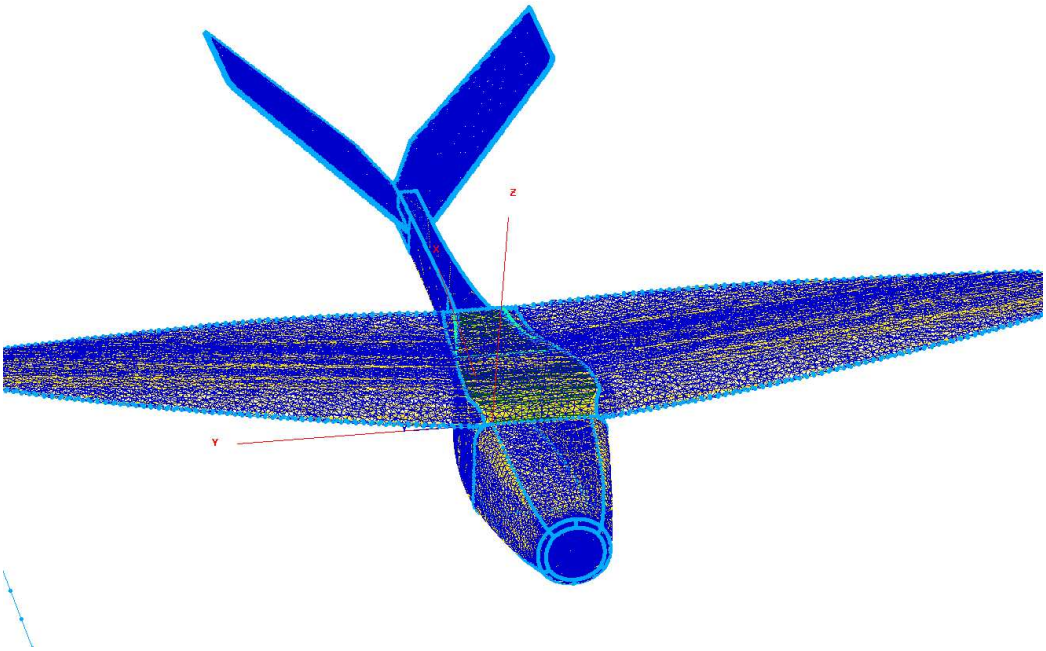

Figure 14: CFD surface grid edges

This is a cost for any analysis using unstructured grids, but becomes particularly costly in this work since the splines connecting the movement of the structure and CFD grids scale directly with the number of surface points. For these grids the MAV surface has on the order of 95k faces and 47k points. The splines connect these fluid grid components to the 255 points of the structure "grid".

3.2 Steady laminar solutions

This series of solutions is an alpha sweep of steady, non-deforming solutions at 10, 20 and 30 mph inlet velocity. The free-stream properties of the cases are shown in Table 4. The alpha sweep includes -10, -5, -2, 2, 4-13, 15, 18, and 20 degrees angle of attack. The three tunnel inlet velocities of 10, 20, and 30 mph were chosen to match the nominal inlet conditions of the previous wind tunnel investigation. The agreement with the experimental data is slightly improved over the previous investigation in (15) by increasing the resolution in the CFD grid near the wall, especially in regions of high curvature.

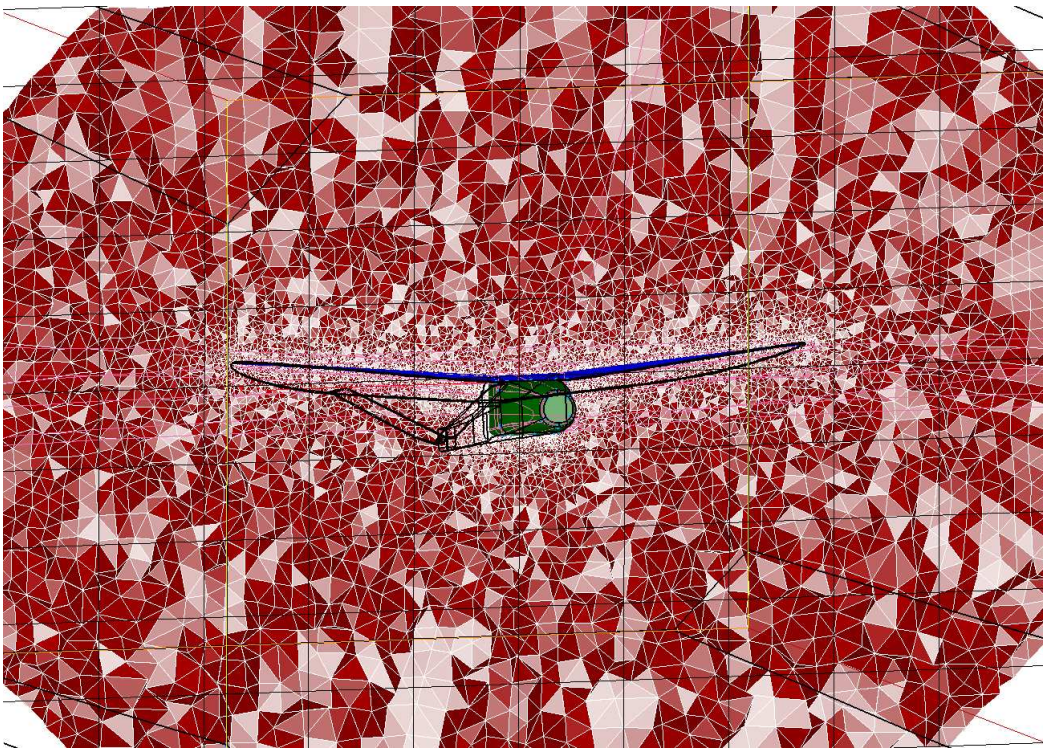

Figure 15: CFD volume grid slice

Table 4: Static solution ranges

	Angle of Attack	Mach Number	Reynolds Number
Min	-10	1.3×10^{-2}	4.326×10^{4}
Max	20	5.2×10^{-2}	1.730×10^{5}

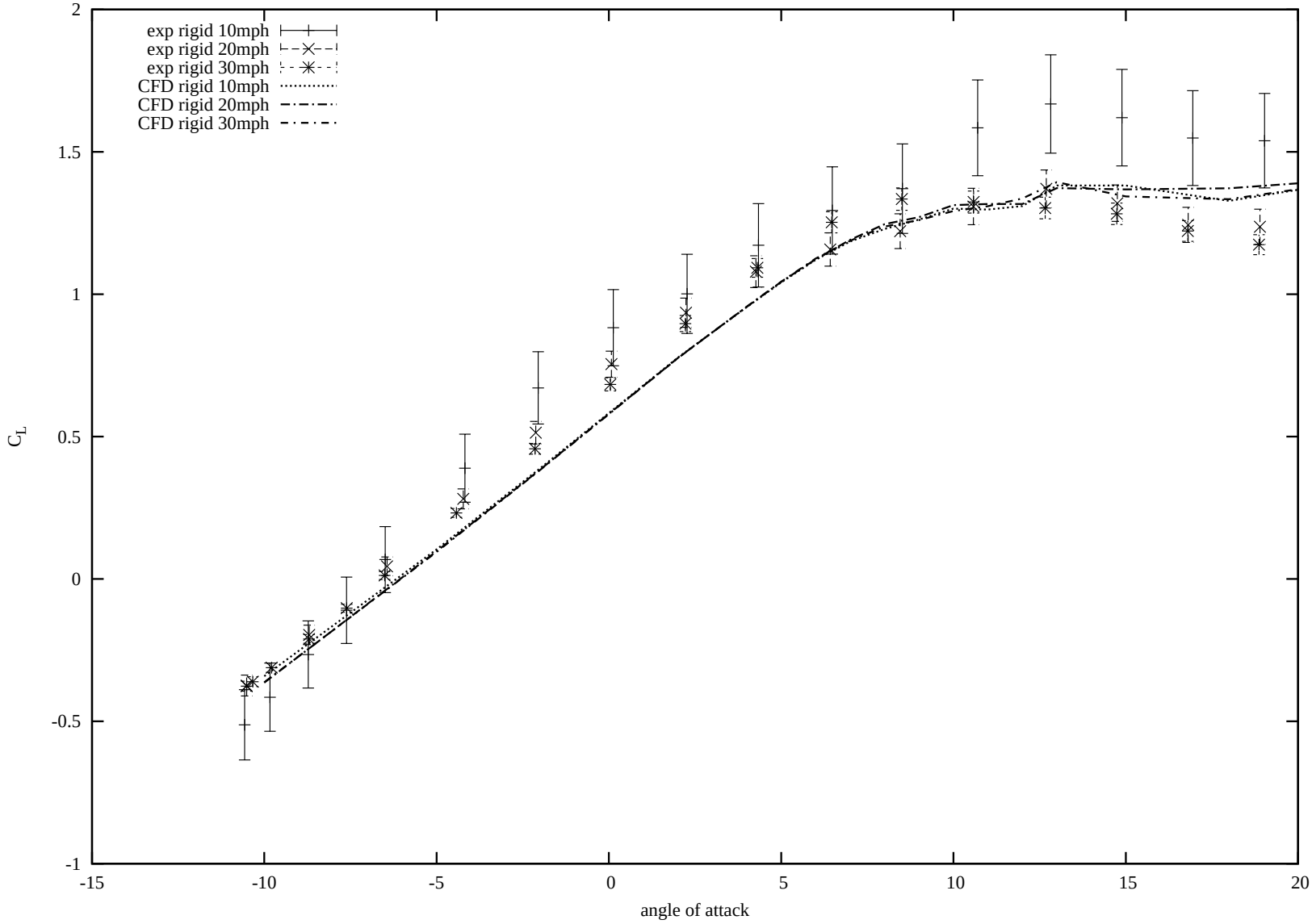

Figure 16: Lift coefficient variation with angle of attack - steady, laminar

The lift coefficient for steady-state laminar solutions with the MAV in the tunnel is shown in Figure 16 . These solutions were run with all rigid surfaces and the second order upwind momentum discretization, and compare well to the experimental data for the rigid MAV from (7). The Reynolds number is low enough that a completely laminar solution can be safely assumed up until significant separation occurs. The onset of stall seems to be predicted well by the CFD for the rigid case.

For the rigid wing MAV, the experimental results show no measurable variation with velocity in the linear region of the lift curve (note overlapping error bars in Figure 16). The non-deforming CFD solution also shows this lack of variation with velocity, and predicts force coefficients closer to the higher velocity wind tunnel data. The 20 and 30 mph cases run in the wind tunnel show the expected Reynolds number variation in stall angle of attack: the higher Reynolds number case separates at a higher angle of attack. The experiments measured the 10 mph case to have a delayed stall angle of attack compared to the two higher Reynolds number cases, which is contrary to the behavior generally expected of stalling airfoils/aircraft. As mentioned previously, the experimental data shown in Figure 16 has been adjusted slightly for wall effects.

As the MAV stalls with increasing angle of attack the measured lift and the computed lift agree closely for the rigid case, up to massive separation around 13 degrees angle of attack, at which point the laminar CFD solution is unsuited to modeling the detached, turbulent eddies and free-shear layers off the suction surface of the vehicle.

At 13 degrees angle of attack the CFD solutions show a slight bump in the lift, but at this point the flow is massively separated over large sections of the wing. Figure 17 shows velocity vectors over the wing on a plane 0.15 m from the centerline of the vehicle. The laminar solution is unreliable for this type of flow-field involving large turbulent free-shear layers.

Figure 18 shows the drag coefficient for the non-deforming CFD solution and the rigid wing wind tunnel results. The computation agrees well with the experimental results in the lower portion of the "drag bucket" where skin friction (wall shear) dominates the drag. This is another indication (aside from the low Reynolds number) that the solution is predominantly laminar, since the wall shear is much greater for a turbulent solution.

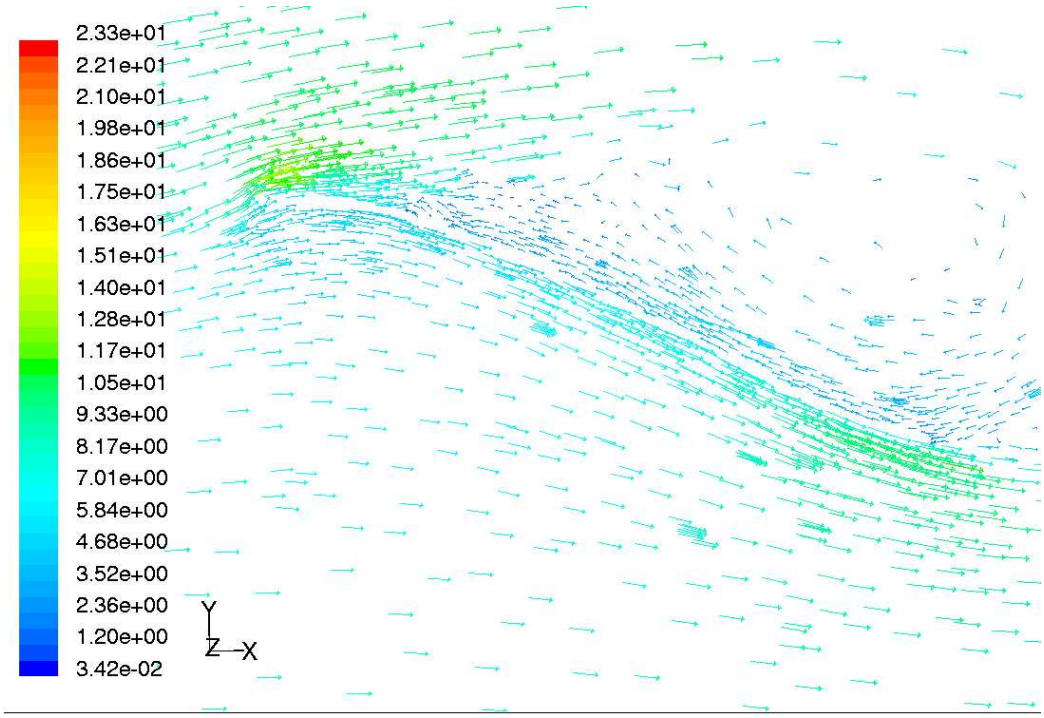

Figure 17: Velocity vectors at 50% span - 13 degrees aoa, 20mph

Though there is negligible difference in the integrated coefficients in the linear region for the different velocities, there is a detectable difference in the flow structure. Figure 19 shows a slight laminar separation bubble develops around 60% of the chord for the 9 degrees angle of attack, 10 mph case, which is not present in the higher velocity case (Figure 20). The bubble is small and static enough that the steady state solution still converges for these low velocity runs. The effect of this bubble was postulated to cause the waviness in the lift coefficient for the 10 mph wind tunnel data. The CFD solutions do not tend to support this, given the size of the bubble, which quickly proceeds to full separation over the rear portion of the wing. The more likely explanation for the odd low speed data is that the forces on the MAV are on the order of the balance resolution for this low velocity. Some waviness is visible in the stall region of the lift curve for the 20 and 30 mph cases, but these cases do not show the separation bubble that is apparent in the low speed case. (7).

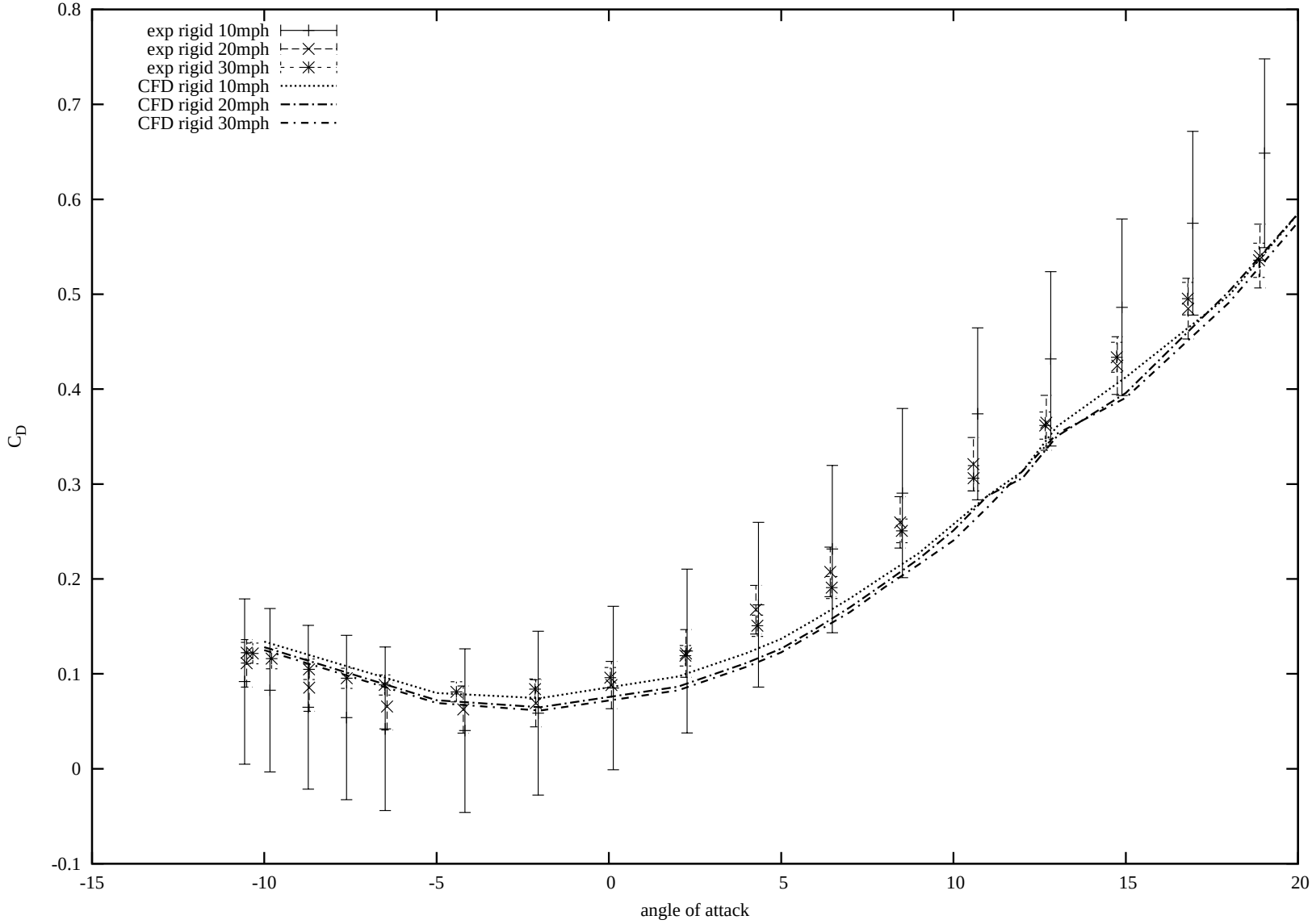

Figure 18: Drag coefficient variation with angle of attack - steady, laminar

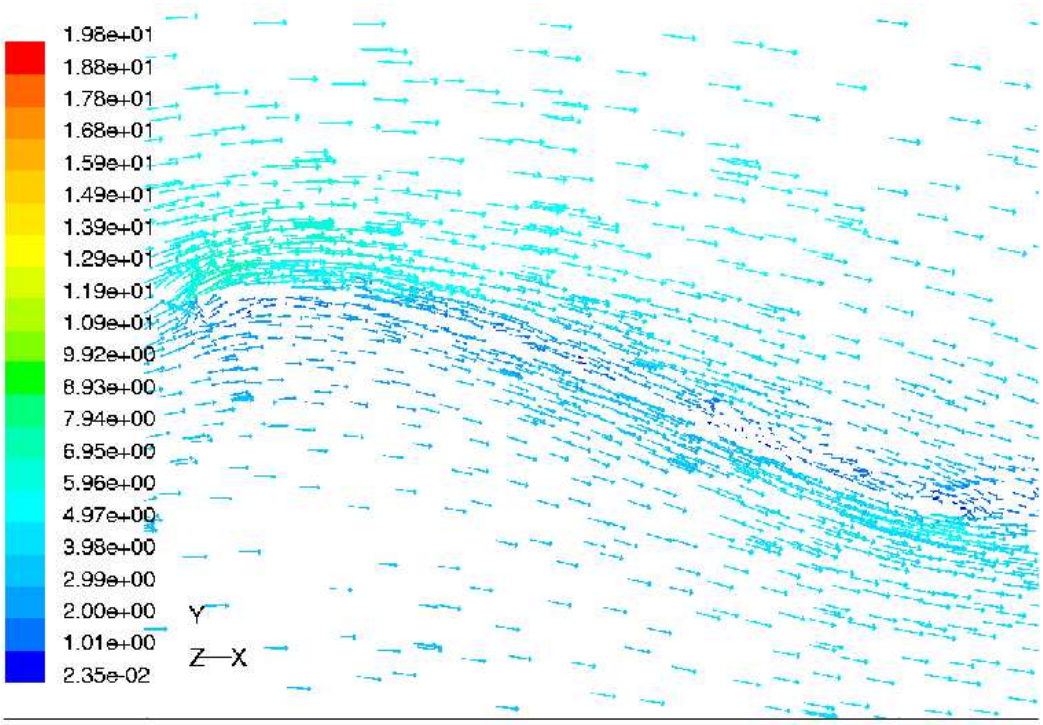

Figure 19: Velocity vectors at 50% span - 9 degrees aoa, 10mph

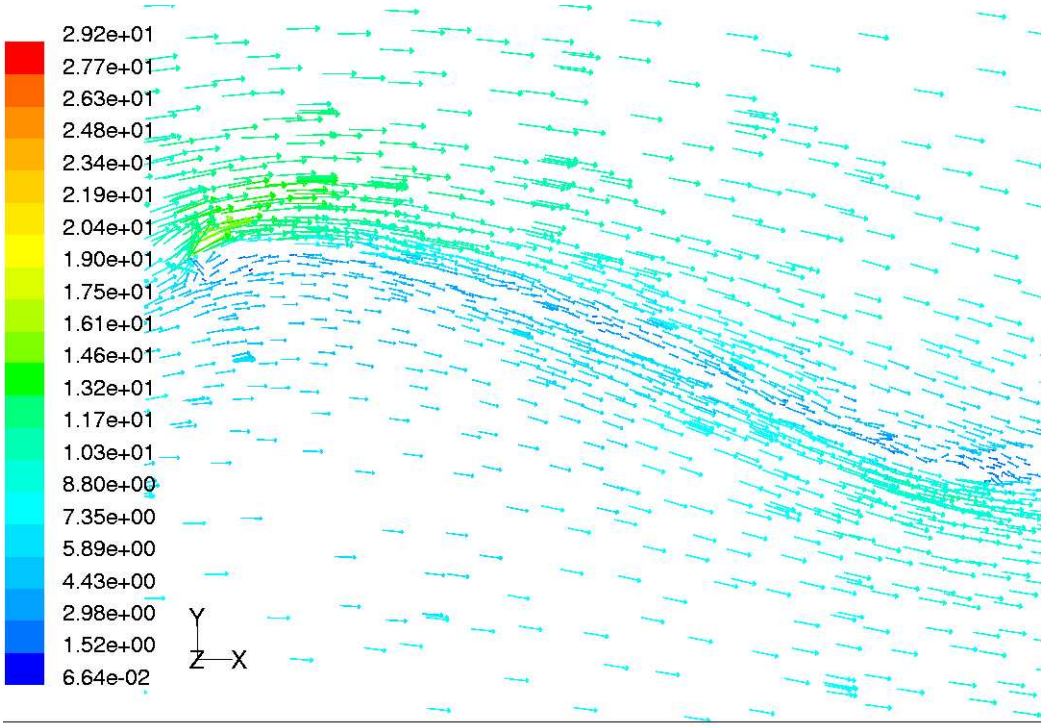

Figure 20: Velocity vectors at 50% span - 9 degrees aoa, 20mph

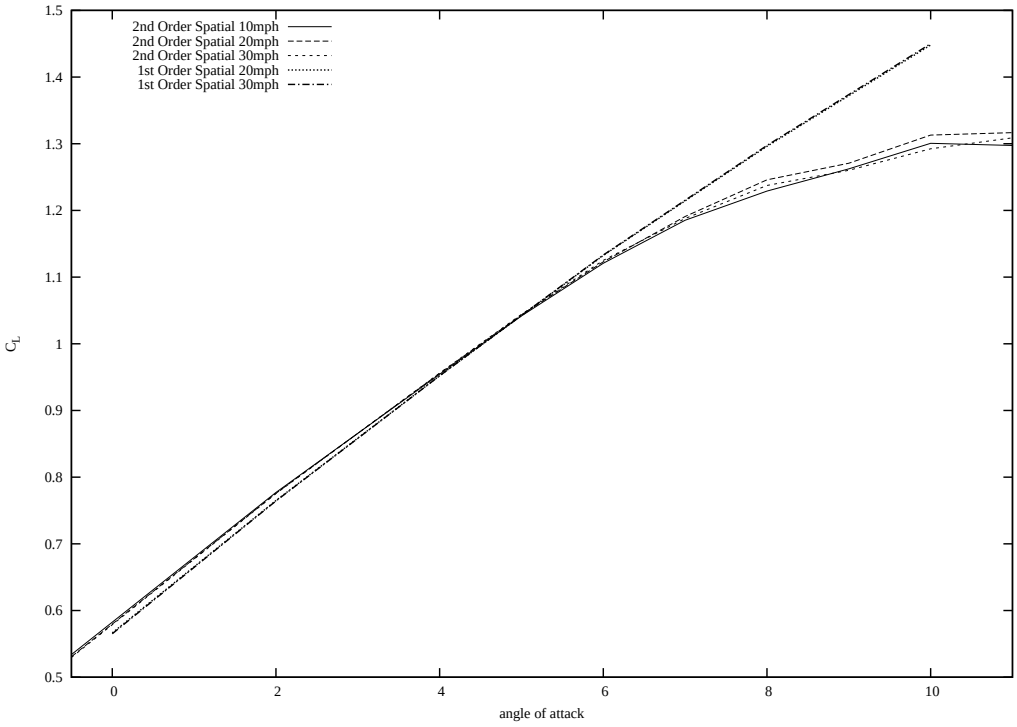

Figure 21: Lift coefficient – 1^{st} and 2^{nd} order spatial discretizations

The effect of the spatial order of accuracy is most clearly visible in the failure of the first order scheme to predict the correct stall angle of attack. Figure 21 shows the continued linear increase in lift between eight and ten degrees angle of attack when the second order solution and experimental results show significant stall behavior.

3.3 Statically Deformed Structure

All of the solutions in this section use the first order upwind discretization for the momentum equations in the fluid solver. This limits the applicability of the model to the linear region of the lift curve, as shown previously by the non-deforming calculations.

The time derivative in the structural model is neglected for these solutions to calculate the static deformations in steady flight conditions. The change in lift due to the static deformation of a membrane wing results from a balance of two competing types of displacements. The trailing edge of the wing can move up higher than the leading edge, which reduces the effective angle of attack ("washout"), and contributes a reduction in lift.

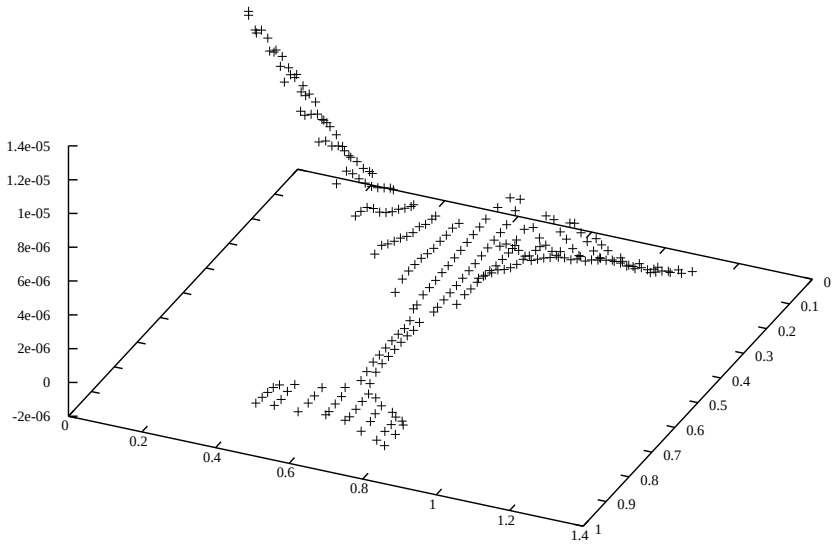

Figure 22: Static deformation – 10aoa, 20 mph

A membrane wing may also "billow" in the wind, which changes the camber of the wing, and can contribute an increase in lift (20).

The deformed shape of the structure at 10 degrees angle of attack is shown in Figure 22. The largest amplitude is calculated for the second mode shape (symmetric bending) over the full angle of attack range tested for the statically deformed MAV. Since the loading at this flight condition is symmetric, this is the mode that should be most excited. Figure 23 shows the amplitude variation with angle of attack for the 20 mph case. One troubling result is the significant amplitude of the first mode shape (anti-symmetric), which is unexpected for this symmetric loading situation. The modal amplitudes increase almost linearly with angle of attack, as would be expected since the lift curve is linear in this region, and the modal amplitudes are a linear function of the force (see Figure 23). The modes retain approximately the same relative magnitude to each other throughout the angle of attack range, with the higher frequency modes being only slightly activated. Even the largest deformations are only on the order of 10^{-5} m.

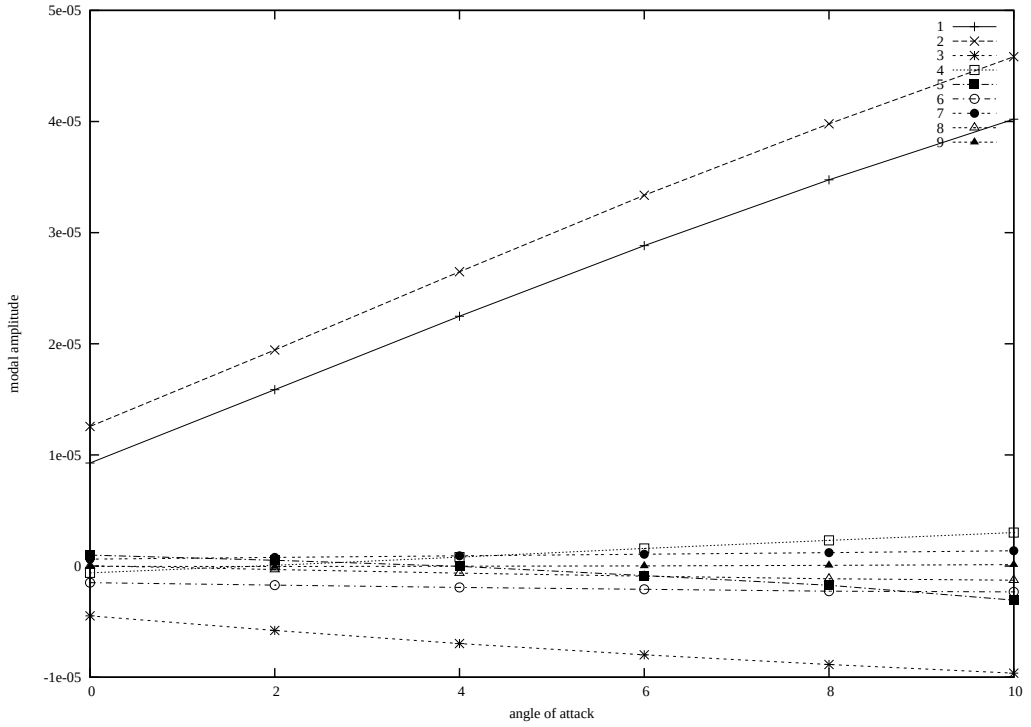

Figure 23: Static modal amplitudes – 20 mph

The significant activation of the anti-symmetric mode is likely the result of the orthonormalization performed on the measured mode shapes. The modes were all scaled to unit length regardless of their measured amplitudes and a uniform mass distribution was assumed. This gives greater displacement magnitudes in the structure due to the loading than would be present in the actual structure with a more physiclly accurate mass matrix. The loading has a significant anti-symmetric component, but it is given too much weight by the present mode normalization.

A more accurate mass matrix could be created by some type of area weighting of the mass of each point, or a reference displacement of the actual MAV could be used to scale the modes to achieve the correct displacement magnitudes in the computational model.

Figure 24 shows the lift calculated using a statically deformed structure, along with the wind tunnel measurements for the flexible wing MAV. The computations do not show the velocity dependence in the lift curve that the wind tunnel experiments measured. Figure 24 shows that the lift computed for 20 and 30 mph free-stream velocities are nearly identical. In the linear region of the lift curve ($\alpha < 8°$) the computed solution falls within the error

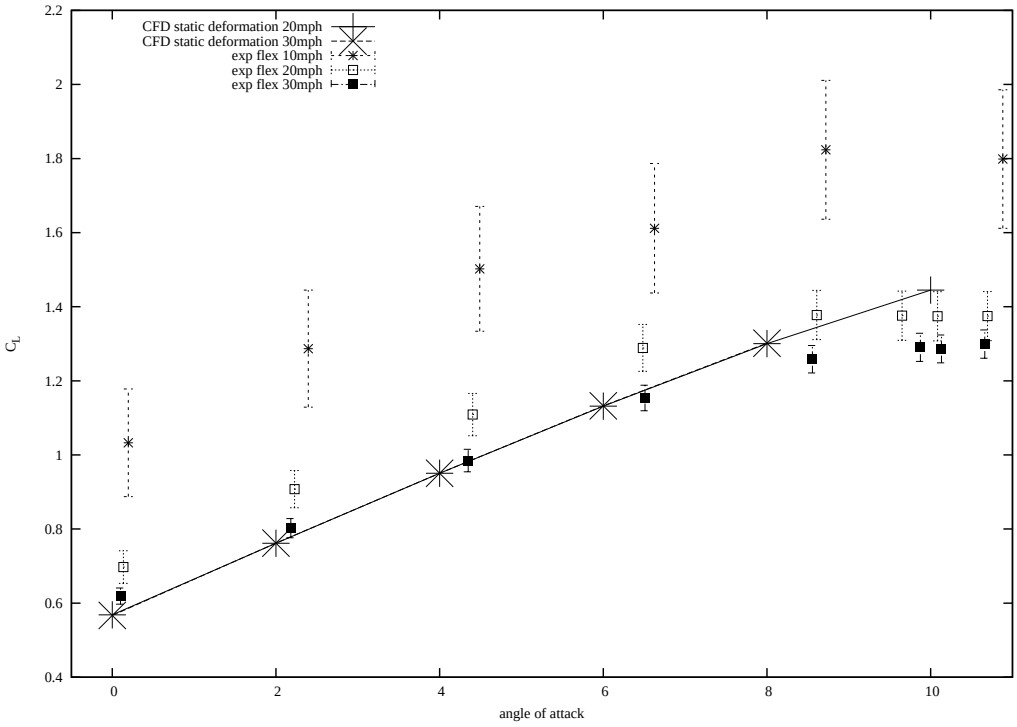

Figure 24: Lift coefficient – statically deformed structure

bars of the 30 mph case, but fails to capture the stall past this angle of attack. This failure was shown for the non-deforming case as well.

The lift computed for the statically deformed structure is nearly identical to that computed for the rigid structure. Figure 25 shows the nearly indistinguishable performance predicted by the rigid and flexible models. This behavior is somewhat supported by the experimental data. Figure 26 shows the overlapping error bars of the rigid and flexible wing for the 10 and 20 mph cases in the linear region of the lift curve. The 30 mph case has a measurable difference in lift even in this region that the computations do not capture.

3.4 Steady Free-stream - Dynamic Structure

The solutions in this section use the first order upwind discretization for the fluid momentum equations and the two previously discussed structural integration schemes. Previous studies of membrane wings have shown "self-initiated" high frequency vibrations in steady flight conditions. This type of behavior is evaluated with the fully dynamic structural model. The time step size for the dynamic solution must be chosen based on the

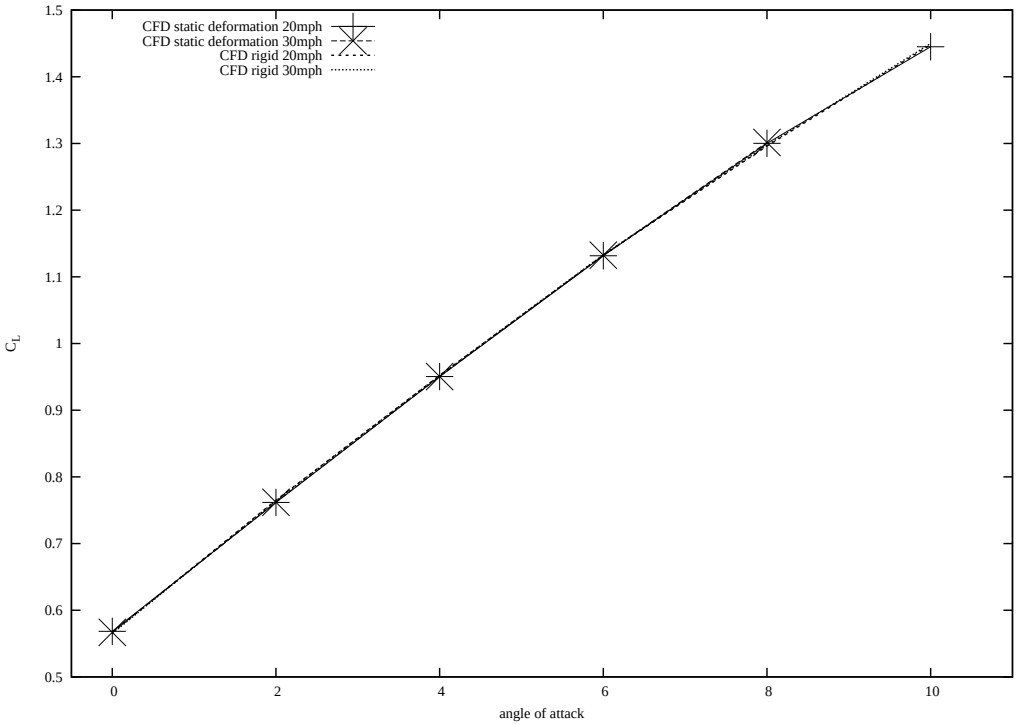

Figure 25: CFD C_L – Rigid and statically deformed MAV

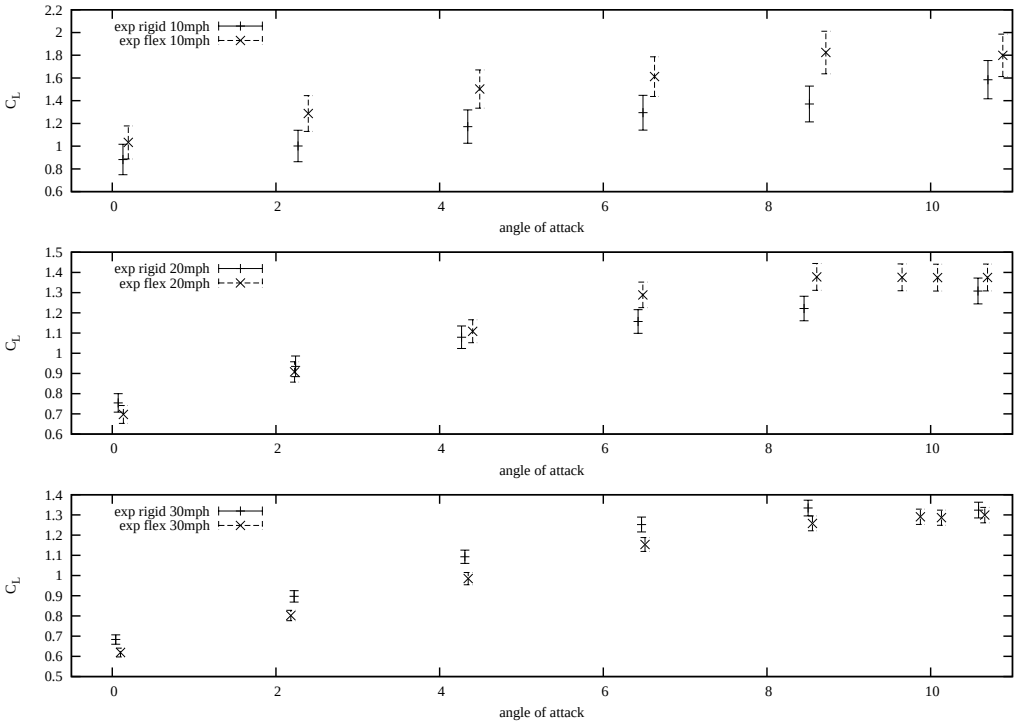

Figure 26: Experimental C_L – Rigid and statically deformed MAV

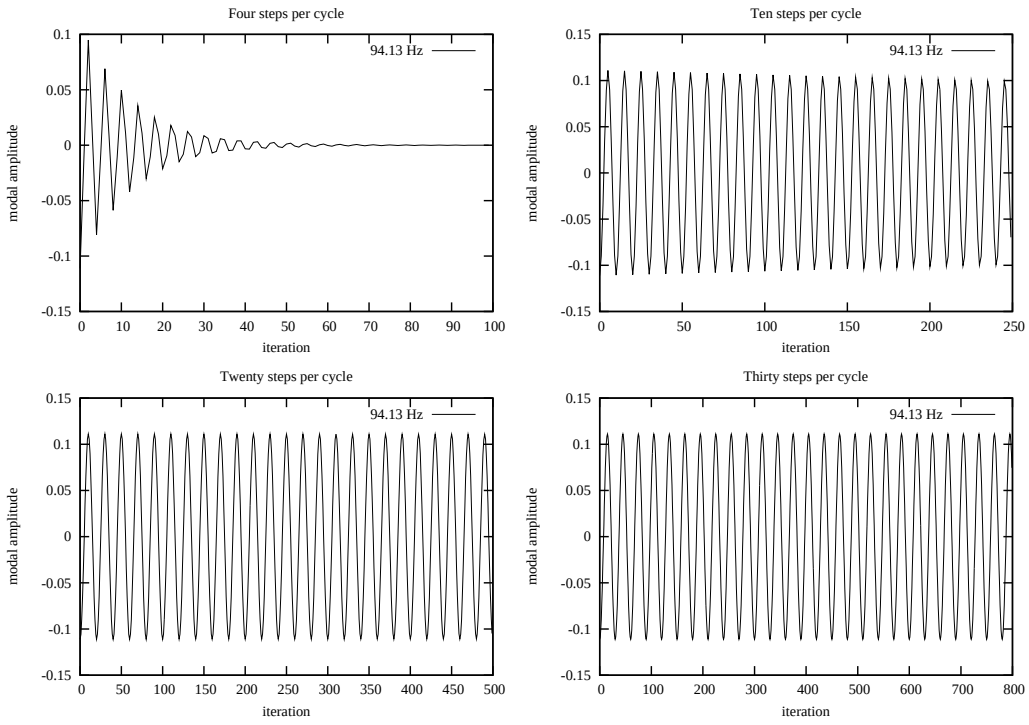

Figure 27: Runge-Kutta – $\mathcal{O}\left(\Delta t\right)^2$

highest frequency of the retained modes, which in this case is the ninth mode at 105 Hz. Figure 27 shows the response of this mode to a displacement perturbation with a time step of $2.6559 \times 10^{-3} s$, or about 4 steps per cycle. The numerical error with a time step this large causes artificial damping and the contribution from this mode is nearly non-existent after only 50 iterations. Even with ten steps per cycle ($\Delta t = 1.0624 \times 10^{-3} s$) the numerical damping of the highest mode is slightly noticeable. With twenty time-steps per cycle of the highest mode and higher the numerical damping is negligible.

Figure 28 shows the response of the ninth mode to an initial displacement perturbation with no external forces or damping integrated by the Newmark method. Even with only four time steps per cycle the Newmark method is able to integrate the mode without undue dissipation, though there are noticeable variations in the amplitude at this resolution. At higher resolutions the two schemes behave similarly.

The ability of either scheme to produce accurate results in the coupled fluid-structure solution is more restricted than the simple structure integration previously presented would indicate. At a resolution of approximately four time steps per cycle of the highest frequency

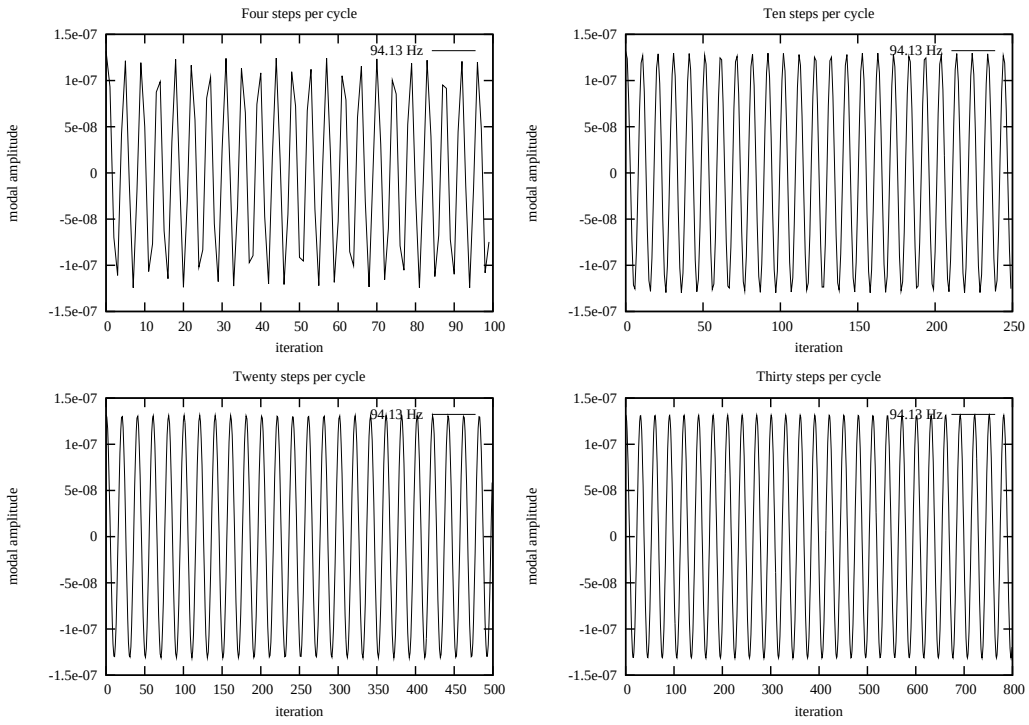

Figure 28: Newmark method - $\mathcal{O}\left(\Delta t\right)^2$

retained, which gives around twenty samples in the lowest frequency, the explicit scheme shows severe dissipation (Figure 27), but the Newmark method showed some promise (Figure 28) at capturing the response in the unforced structure integration. The power spectrum of lift is shown in Figures 29 and 30 for a time step of 0.002 seconds, which gives approximately five time steps per cycle of the highest frequency retained in the structural model. Figure 29 shows the results obtained with the Runge-Kutta scheme, and Figure 30 shows the results obtained with the Newmark method, starting from a statically deformed solution. The vertical lines on the plots are the frequencies of the modes in the structural model. Neither scheme is able to accurately resolve the high frequency modes as in Figures 33 and 34.

A spurious low-frequency component is evident in both cases, most likely caused by aliasing of fluid physics at frequencies higher than 250 Hz (since the time-step is below the Nyquist limit of the structure's frequencies). The explicit scheme has little more than noise at the intermediate to high frequencies, while the implicit scheme produces a significant component (at about 95 Hz) below the highest frequency. An additional spike is also seen

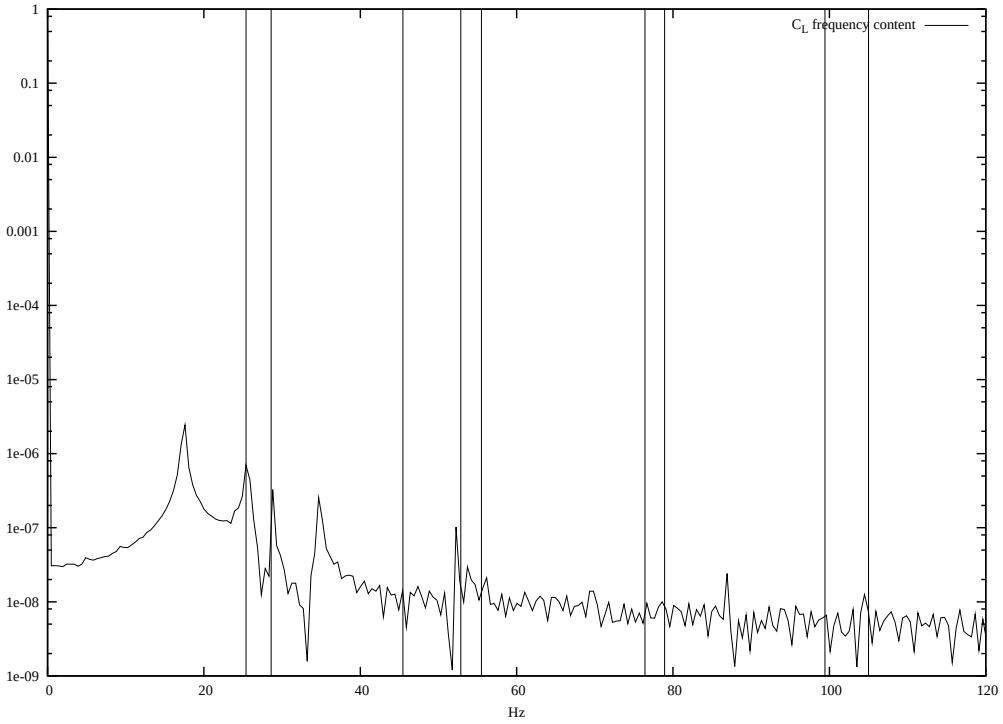

Figure 29: C_L frequency – 10aoa, $\Delta t = 0.002$, Runge-Kutta

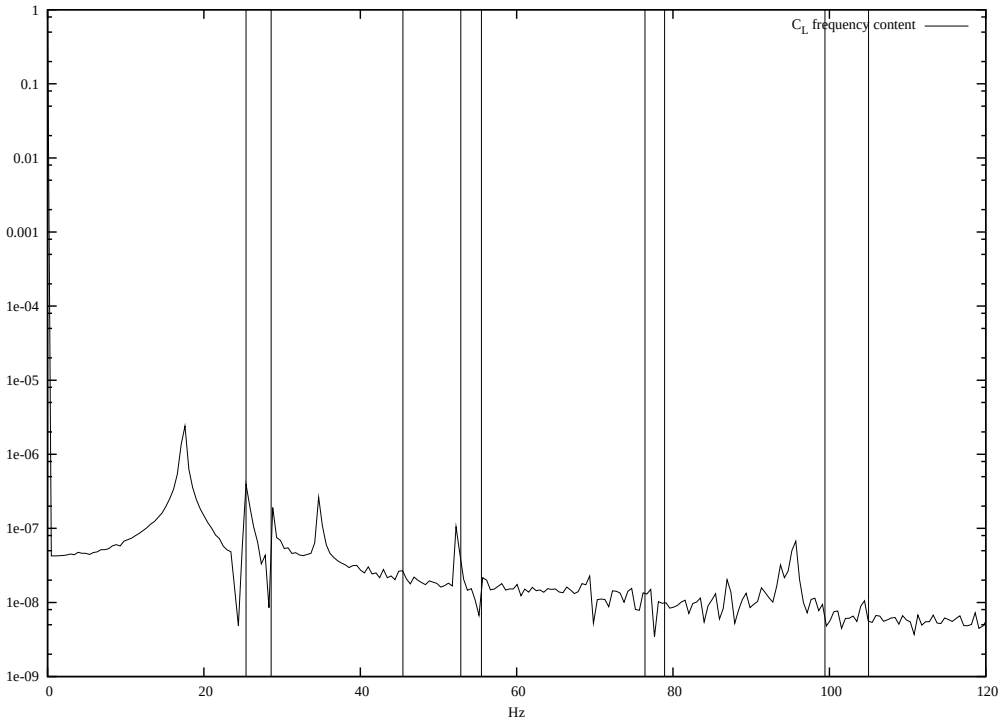

Figure 30: C_L frequency – 10aoa, $\Delta t = 0.002$, Newmark

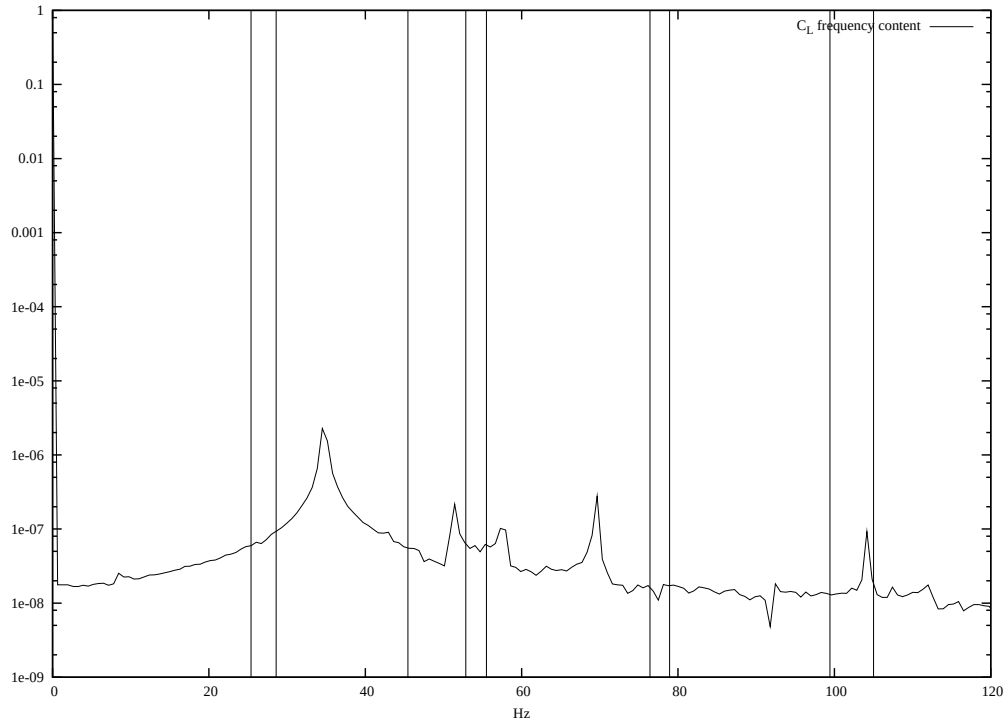

Figure 31: C_L frequency – 10aoa, $\Delta t = 0.001$, Runge-Kutta

in both methods between the second and third mode, the Newmark method calculates this component to be larger than the component at the second mode shape.

Reducing the time-step to 0.001 s does not noticeably improve the performance for either scheme. Figures 31 and 32 show the lift variation, which is completely different from the resolved solution at $\Delta t = 0.0005$s (shown in Figure 33). The aliasing actually seems to be worse in this case than the solution with $\Delta t = 0.002$s. As mentioned previously, this is most likely an effect of the non-linear fluid physics being under-sampled.

The variation of lift coefficient in time beginning from a statically converged solution is shown in Figure 33. This solution is integrated with the explicit scheme examined in section 2.3.2, using a time step of 5×10^{-4}s, which is approximately twenty time steps per cycle of the highest mode. The sharp spikes are due to restart errors with FLUENT, but they are much higher frequency than the relevant variations due to the physics. Figure 34 shows the power spectrum of the lift coefficient up to 120 Hz. The vertical lines on the graph mark the natural frequencies of the modes. At this time step we see significant peaks at the frequencies of the structural model, which is the expeected behavior. This shows that

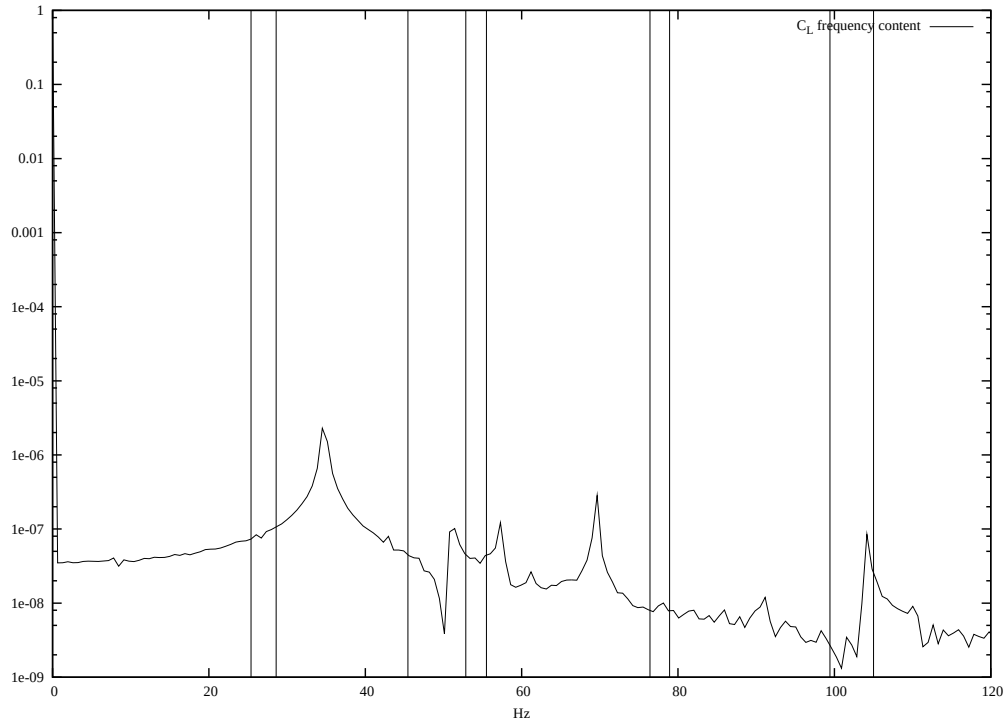

Figure 32: C_L frequency – 10aoa, $\Delta t = 0.001$, Newmark

the characteristic frequencies of the fluid behavior are significantly higher than the structure (since the structure is resolved at the previously presented larger time-steps). Surprisingly, significant power is associated with the higher frequency mode around 100 Hz. These high frequency fluctuations, similar to those reported in (11), are termed "self-initiated", and are due to small velocity and pressure perturbations present in any real flow, or roundoff errors present in a computational flow. The effect of large flow disturbances is further evaluated in Section 3.5.

The frequency adjustment made to remove the affect of ambient pressure from the measured mode shapes shifted the frequencies higher than those measured at atmospheric pressure. The frequencies of the response would be expected to move back near their measured values in the aeroelastic solution, but they actually have a tendency to shift higher than the frequencies in the model. This is especially visible in the higher modes, shown in Figure 34.

The non-linear coupling between the modes in the fluid-structure interaction requires that all of the modes be accurately resolved. Higher resolution than would be expected

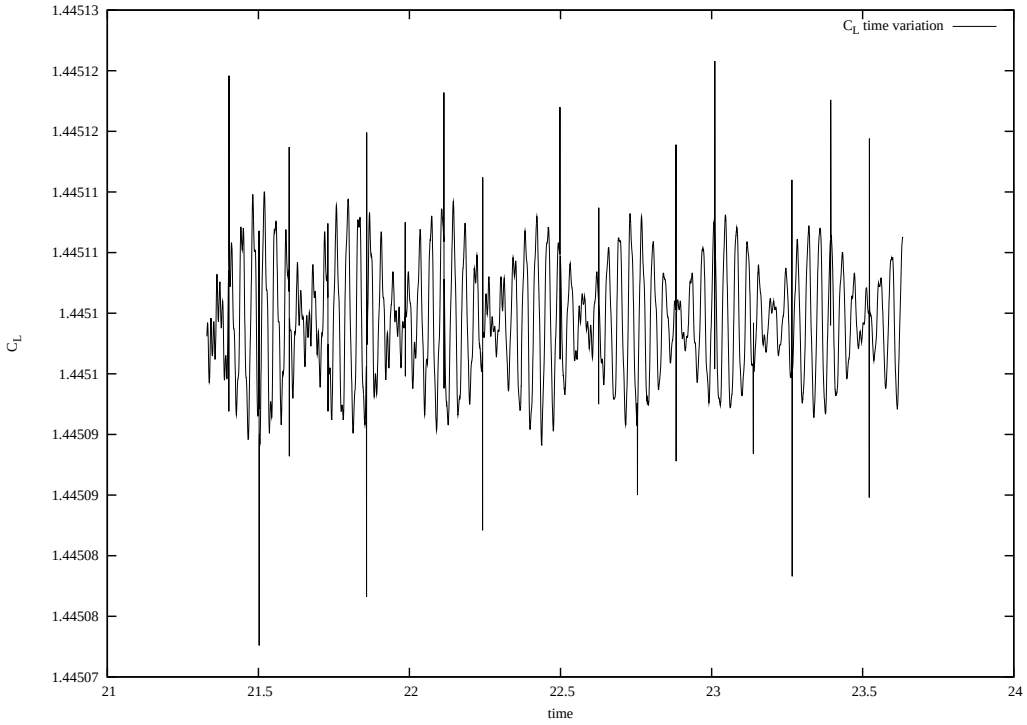

Figure 33: C_L time – 10aoa, $\Delta t = 0.0005$, Runge-Kutta

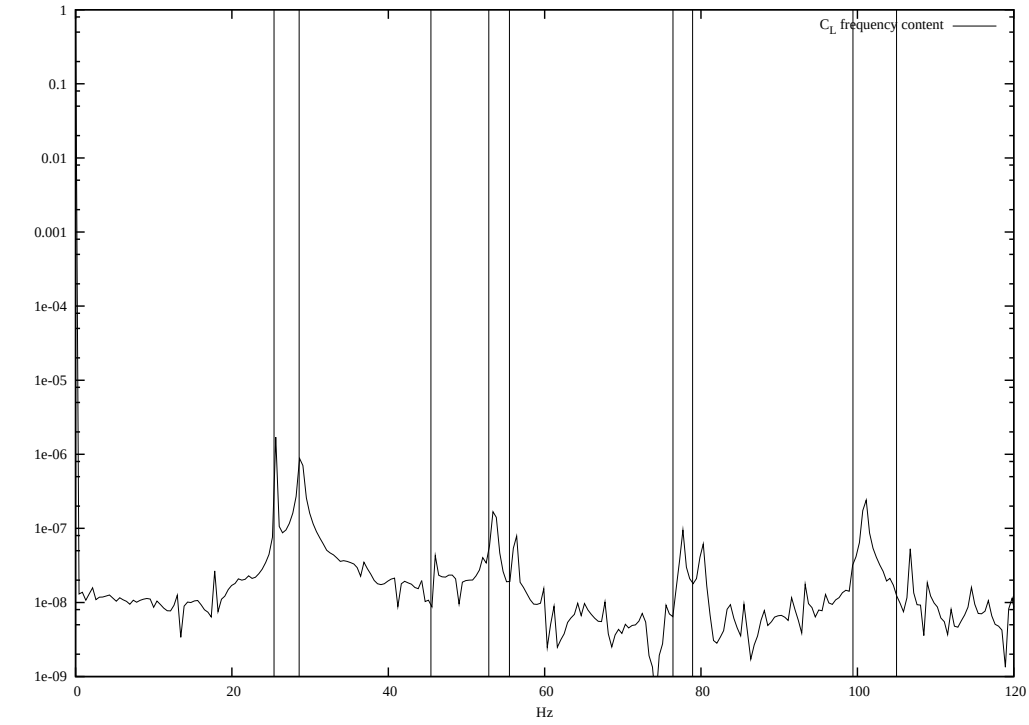

Figure 34: C_L frequency – 10aoa, $\Delta t = 0.0005$, Runge-Kutta

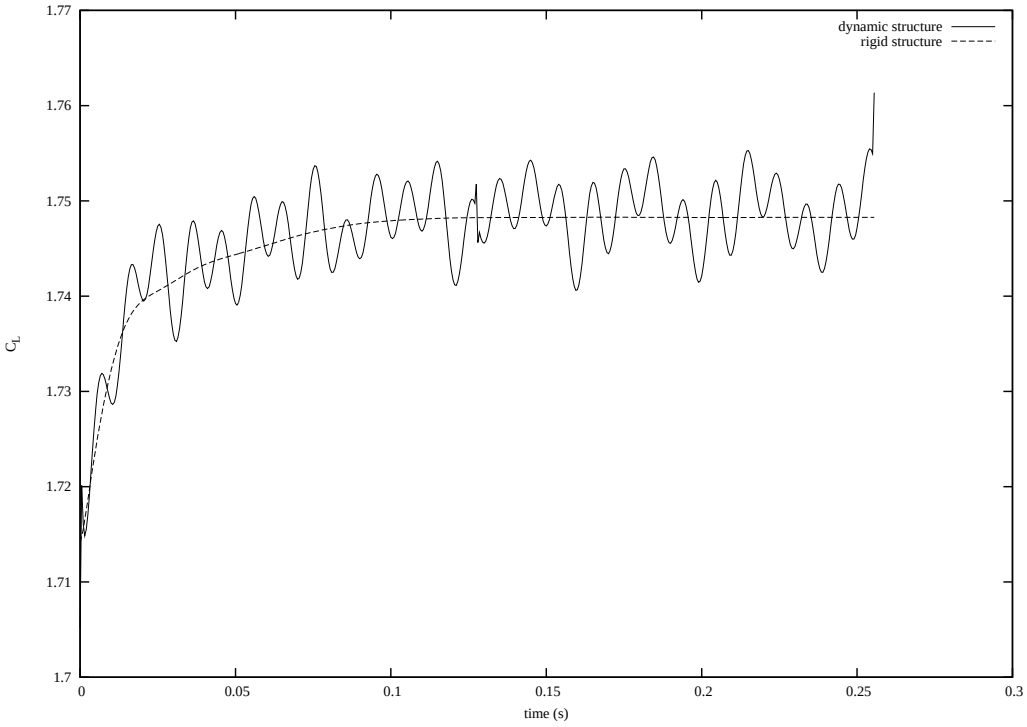

Figure 35: 10% Gust time response – step input, 10aoa

from looking simply at the decoupled structure (Figures 27 and 28) is required in the closely coupled algorithm.

3.5 Unsteady Gust Response

The aeroelastic solution method is used to evaluate the MAV's gust response and unsteady performance capabilities, as well as the effectiveness of the adaptive washout mechanism at stabilizing lift production. The restart errors in FLUENT, which were only a nuisance in the previous analysis, are now more troublesome, since the solution must be restarted after applying a gust at the inlet. For this reason the first two time-steps after the gust are not shown in the subsequent figures.

Figure 35 shows the response of the flexible and rigid MAV in a free-stream of 8.9408 m/s and 10 degrees angle of attack to a ten percent step increase in the free-stream velocity. The dynamic structure does not provide any noticeable smoothing of the lift increase over the response of the rigid structure. The high frequency components of the structure are excited, as shown in Figure 36. It is understandable that the high frequency modes of the

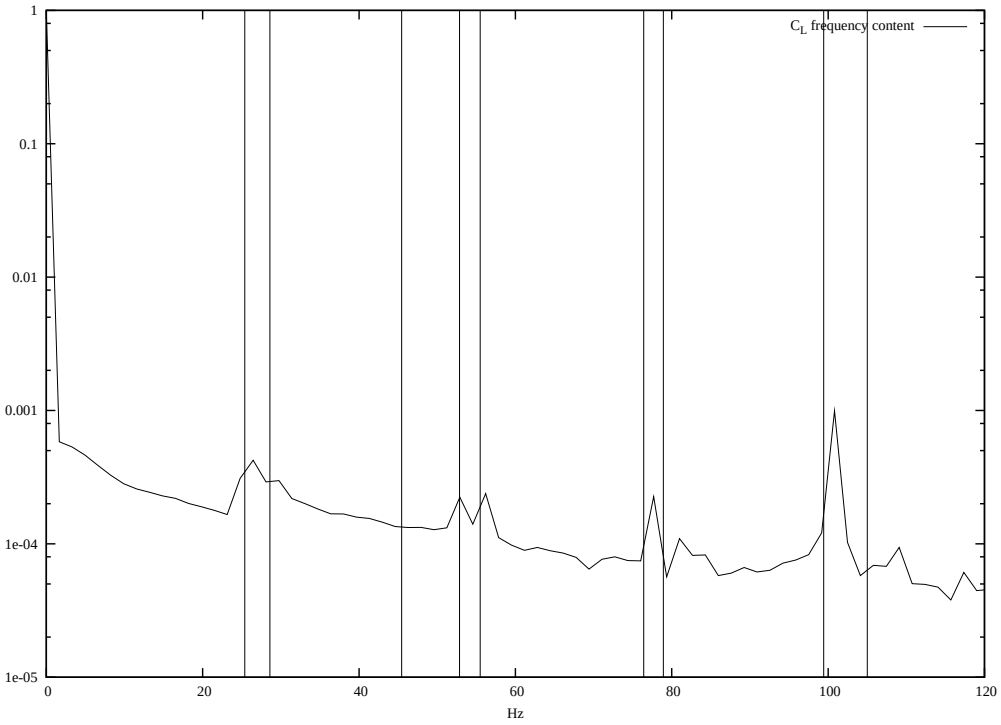

Figure 36: 10% Gust frequency response (dynamic structure) – step input, 10aoa

MAV would play an important role in the vehicle's gust response, since a step input can be thought of as a forcing function at infinite frequency.

Since the MAV will experience large gusts as part of its normal operations the high frequency vibrations of the membrane wing are significant to its performance. As Figure 36 shows, the high frequency modes are the most significant part of the lift variation in the gust response.

IV. Conclusions

The aeroelastic solution method presented in this thesis is capable of modeling the primary behavior of the membrane wing aerodynamics. The steady, non-deforming laminar solutions predict the MAV's stall angle of attack when a 2nd order upwind scheme is used, but fail to match the wind tunnel data with a first order scheme. The dynamic performance of the MAV shows significant high frequency activity even under steady flight condition.

This flexible wing MAV shows mixed results in achieving all of the claimed advantages of membrane wings. The stall performance of the vehicle can not be properly evaluated with this model, since the first order discretization fails to capture the stall characterisitcs of the non-deforming case. The "adaptive washout" mechanism seems to provide no smoothing influence on the lift production of the flexible wing over the rigid wing in a gust.

4.1 Evaluation of the Model

The modal model predicts the proper linear variations in modal amplitudes with increasing MAV angle of attack, but the actual size of the wing displacements is sensitive to the normalization chosen for the mode shapes. The choice of normalization has a direct scaling effect on how the forces translate to displacements in the model, and is shown to be an area in need of future improvement and validation.

The gust response is difficult to evaluate accurately with the present model for a variety of reasons. The scaling of the modes would have an affect on the time response, and the poor restart capabilities of the fluid solver make the first two iterations after a restart unreliable.

4.2 Design Considerations

The performance of the MAV (both computational and experimental) indicates that changes to the wing to promote "billowing" type deformations rather than "washout" type deformations could improve the lift production over the rigid wing in the linear region of the lift curve. The current flexible wing design slightly under-performs the rigid wing at

nominal angles of attack (for those velocities where there is a measurable difference), but shows good resistance to stall over the rigid wing due to the static washout.

The unsteady results indicate that the wing is too stiff to provide any smoothing in gusty conditions, the dynamic flexible structure response being nearly as step-like as that of the rigid structure. Other MAV's that have been reported to enjoy this smoothing effect had latex rubber as the inter-spar material rather than parachute nylon, which might have contributed to the desirable lift smoothing.

4.3 Considerations for Further Analysis

The basic method presented shows promise, but there are significant areas for improvement in this aeroelastic model. Both the fluid and structural components have significant shortcomings. The scope of the method's applicability can also be extended by improving the fluid solver to handle unsteady detached flows. Also, further insight into the effect of variations in inter-spar material would benefit future MAV designers.

There are two approaches to improving the shortcomings in the structural model. The first would be to keep the uniform mass matrix, and apply some type of scaling to the mode amplitudes based on the measured response of the MAV. The second is keeping the present normalization and building a more exact mass matrix based on physical measurements or more detailed models of the vehicle.

In the first approach each mode could be normalized by a ratio of its amplitude to the largest mode's amplitude. A scaling by the ratio of ω^2 of the mode to the largest mode's ω^2 could be used to correct for the effect of larger velocities at higher frequencies giving larger measured amplitudes by the laser vibrometry. This gives an effective "modal mass" (4). The problem with this approach is the unspecified proportionality of a unit displacement of the largest mode to a the magnitude of force in that mode. An alternative would be constructing a more exact mass matrix, since this analysis assumed a uniform mass distribution. The assumption is not greatly detrimental to the static deformations in this case since the wind tunnel data showed them to have only a small effect on the lift. However, the effectively increased inertia of the wing (which is actually lighter than the rest of the aircraft) will cause the unsteady response of the structure to be slower due to the

increased mass to stiffness ratio. An improvement might involve some type of area average along with a weighting based on what parts of the MAV the measurement point is near (membrane, spar, leading edge, etc), or a more detailed physical model, such as a finite element model. The utility of the model could be extended most directly by allowing for sub-iteration updates (no time lagging) of the force transformation between the fluid and structure. Extension to higher order spatial accuracy on the moving grid would also be advantageous.

The fluid solver needs to be capable of higher order spatial accuracy to capture the relevant flow physics while maintaining a reasonable problem size. The size of the splines used to couple the fluid and structure are driven by the number of points on the vehicle's surface, and can quickly make doing extensive analysis prohibitive due to hardware limitations.

More detailed analysis of the coupling (if any) between flow separation, shedding and the membrane vibrations can be performed by using a detached eddy simulation (DES) for the flow physics. The current methods being applicable only to mostly attached laminar flows.

The design of future MAV's would also benefit from an analysis of how different types of inter-spar material affect the dynamic response of the wing, most importantly, their effect on any desirable passive control properties such as lift smoothing in a gust response.

Bibliography

1. Bisplinghoff, Raymond L., et al. *Aeroelasticity*. Mineola, NY: Dover Publications, Inc., 1996.
2. Blazek, J. *Computational Fluid Dynamics: Principles and Applications*. Elsevier, 2001.
3. Cobb, Richard. "personal communication." disscusion of experimental results in vacuum, Oct 2004.
4. Cobb, Richard. "personal communication." disscusion of computational results and laser vibrometry measurements, Feb 2005.
5. Cobb, Richard and Kevin Ponzianni. "Ambient Pressure Effects on the Modal Characterisitcs of a Micro Air Vehicle." Unpublished presentation of research results, 2004.
6. Cook, Robert D., et al. *Concepts and Applications of Finite Element Analysis* (Fourth Edition). New York, NY: John Wiley and Sons, Inc., 2002.
7. DeLuca, Anthony. *Experimental Investigation into the Aerodynamic Performance of Both Rigid and Flexible Wing Structured Micro-Air-Vehicles*. MS thesis, Air Force Institute of Technology, 2004.
8. Duchon, Jean. "Splines Minimizing Rotation-Invariant Semi-Norms in Sobolev Spaces." *Constructive Theory of Functions of Several Variables: Proceedings of a Conference Held at Oberwolfach, April 25-May 1, 1976*, edited by W. Schempp and K. Zeller. 85 – 100. Berlin: Springer-Verlag, 1977.
9. Farhat. "CFD Based Simulation of the Unsteady Aeroelastic Response of a Maneuvering Vehicle." Number 2000-0899. 2000.
10. Ifju, Peter G., et al. "Flexible-wing-based Micro Air Vehicles." Number AIAA-2002-0705. AIAA, 2002.
11. Lian, Yongsheng and Wei Shyy. "Three-dimensional Fluid-Structure Interactions of a Membrane wing for Micro Air Vehicle Applications." Number AIAA-2003-1726. AIAA, April 2003.
12. Lian, Yongsheng, et al. "A Computational Model for Coupled Membrane-Fluid Dynamics." Number 2002-2972. 2002.
13. Liu. "Calclations of Wing Flutter by a Coupled CFD-CSD Method." Number 2000-0907.
14. McMichael, James M. and Col(Ret.) Michael S. Francis. *Micro Air Vehicles - Toward a New Dimension in Flight*. Technical Report, DARPA, 1997.
15. M.DeLuca, Anthony, et al. "Experimental Investigation into the Aerodynamic Properties of a Flexible and Rigid Wing Micro Air Vehicle." Number 2004-2396. AIAA, 2004.
16. Parker, Maj Gregory H. "Dynamic Aeroelastic Analysis of Wing/Store Configurations." Disertation Prospectus, July 2004.

17. Polytec. *Theory Manual: Polytec Scanning Vibrometer*, 2004.

18. Strang, Gilbert. *Linear Algebra and its Applications* (3rd edition Edition). Thomson Learning Inc, 1988.

19. Versteeg, H.K. and W. Malalasekera. *An introduction to Computational Fluid Dynamics: The Finite Volume Method*. Prentice Hall, 1995.

20. Waszak, Martin R., et al. "Simulation and Flight Control of an Aeroelastic Fixed Wing Micro Aerial Vehicle." Number AIAA-2002-4875. AIAA, August 2002.

21. Waszak, Martin R. and Luther N. Jenkins. "Stability and Control Properties of an Aeroelastic Fixed Wing Micro Aerial Vehicle." Number AIAA-2001-4005. AIAA, August 2001.

Appendix A. Mass-orthogonality of mode shapes

This derivation follows that presented in Appendix C of (6). The harmonic motion of a structure with no viscous damping is described by

$$\left(K - \omega^2 M\right) d = 0 \tag{48}$$

The i^{th} and j^{th} eigenpair satisfy

$$\left(K - \omega_i^2 M\right) d_i = 0 \tag{49}$$

and

$$\left(K - \omega_j^2 M\right) d_j = 0 \tag{50}$$

respectively. Premultiply 49 by d_j^T and 50 by d_i^T, giving

$$d_j^T K d_i - d_j^T \omega_i^2 M d_i = 0 \tag{51}$$

and

$$d_i^T K d_j - d_i^T \omega_j^2 M d_j = 0 \tag{52}$$

Since both K and M are symmetric

$$\begin{array}{rcl} d_j^T K d_i & = & d_i^T K d_j \\ d_j^T M d_i & = & d_i^T M d_j \end{array} \tag{53}$$

and subtracting 52 from 51 gives

$$\left(\omega_j^2 - \omega_i^2\right) d_j^T M d_i^T = 0 \tag{54}$$

For $\omega_i^2 \neq \omega_j^2$

$$d_j^T M d_i = 0$$

The mode shapes of different frequencies are M orthogonal.

B.1 chkmalloc.c

```
#include <stdio.h>
#include <stdlib.h>
#include "chkmalloc.h"
/* this routine taken from C programming notes by Steve Summit */

void *chkmalloc(size_t sz)
{
void *ret = malloc(sz);
if(ret == NULL)
{
fprintf(stderr, "Out of memory\n");
exit(EXIT_FAILURE);
}
return ret;
}
```

B.2 getwords.c

```
#include "getwords.h"
/* this routine taken from C programming notes by Steve Summit */

int getwords(char *line, char *words[], int maxwords)
{
char *p = line;
int nwords = 0;

while(1)
{
while(isspace(*p))
p++;

if(*p == '\0')
return nwords;

words[nwords++] = p;

while(!isspace(*p) && *p != '\0')
p++;

if(*p == '\0')
return nwords;

*p++ = '\0';
```

```
if(nwords >= maxwords)
return nwords;
}
}
```

B.3 ortho-norm.c

```
/* program to create an orthonormal basis of eigenvectors from
   measured mode shapes */
#include "ortho-norm.h"
#include "getwords.h"
#define MAXLINE 200
#define MAXWORDS 20

int main(int argc, char *argv[])
{/* first argument is a text file with n, and m, and then the xy
    coordinates of the nodes, the second argument is the text file
    with the modes as its columns */
  RKIND **A; /* storage for the modes */
  RKIND **B; /* storage for the amount subtracted off in Gramm-Schmidt */
  RKIND *y; /* y coordinate of nodes */
  RKIND *x; /* x coordinate of nodes */
  RKIND *scale; /* for normalizing each eigenvector */
  RKIND *omega; /* natural frequency */
  RKIND **ortho; /* check for mutual orthogonality of eigenvectors */
  IKIND *n; /* rows */
  IKIND *m; /* columns */
  RKIND *mass; /* total mass of the structure */
  int i;
  int j;
  int k;
  FILE *A_file; /* file with measured shapes as columns */
  FILE *valu_file; /* file with some values in it */
  FILE *ortho_tab; /* output the results */
  FILE *w_file;/* file with the measured freqs in Hz */
  FILE *w_file_out;
  char line[MAXLINE];
  int ac;
  char *av[MAXWORDS];

  n = (IKIND *)chkmalloc( sizeof(IKIND) );
  m = (IKIND *)chkmalloc( sizeof(IKIND) );
  mass = (RKIND *)chkmalloc( sizeof(RKIND) );
  scale = (RKIND *)chkmalloc( sizeof(RKIND) );

  if(argc!=4){printf("Usage:\n      %s value_filename A_filename w_file\n",argv[0]); exit(1)
```

```
else{
  A_file = fopen(argv[2],"r");
  if(A_file==NULL){printf("Can't open %s for read.\n", argv[2]); exit(1);}
  valu_file = fopen(argv[1],"r");
  if(valu_file==NULL){printf("Can't open %s for read.\n", argv[1]); exit(1);}
  w_file = fopen(argv[3],"r");
  if(w_file==NULL){printf("Can't open %s for read.\n", argv[3]); exit(1);}
  w_file_out = fopen("stiffness.dat", "w");
}
/* read in the number of modes, the number of nodes, and the coordinates */
fgets(line, MAXLINE, valu_file);
ac = getwords(line, av, MAXWORDS-1);
n[0] = atof(av[0]);
m[0] = atof(av[1]);
mass[0] = atof(av[2]);
y = (RKIND *)chkmalloc( m[0]*sizeof(RKIND) );
x = (RKIND *)chkmalloc( m[0]*sizeof(RKIND) );
omega = (RKIND *)chkmalloc( n[0]*sizeof(RKIND) );
i = 0;
while(fgets(line, MAXLINE, w_file) != NULL){
  ac = getwords(line, av, MAXWORDS-1);
  omega[i] = atof( av[0] );
  omega[i] = TWO*M_PI*omega[i]*TWO*M_PI*omega[i];
  fprintf(w_file_out, "%1.12e\n",omega[i]);
  i++;
}
i = 0;
while(fgets(line, MAXLINE, valu_file) != NULL){
  if(i<m[0]){
    ac = getwords(line, av, MAXWORDS-1);
    x[i] = atof(av[0]);
    y[i] = atof(av[1]);
  }else{
    printf("Wrong number of nodes. Specified: %i, Reached: %i\n",m[0],i);
    exit(1);
  }
  i++;
}
A = (RKIND **)chkmalloc( n[0]*sizeof(RKIND) );
B = (RKIND **)chkmalloc( n[0]*sizeof(RKIND) );
ortho = (RKIND **)chkmalloc( n[0]*sizeof(RKIND) );
for(i=0; i<n[0]; i++){
  /* A and B are (n x m), ortho is only (n x n) */
  A[i] = (RKIND *)chkmalloc( m[0]*sizeof(RKIND) );
  B[i] = (RKIND *)chkmalloc( m[0]*sizeof(RKIND) );
  ortho[i] = (RKIND *)chkmalloc( n[0]*sizeof(RKIND) );
```

```
  }
  /* the mode file for aeroelastic.c has the eigenvectors as its columns */
  j = 0;
  while(fgets(line, MAXLINE, A_file) != NULL){
    ac = getwords(line, av, MAXWORDS-1);
    if( ac != n[0] ){
      printf("Wrong number of modes found in %s.\n",argv[2]);
      exit(1);
    }else{
      for(i=0; i<n[0]; i++){
A[i][j] = (RKIND)atof(av[i]);
      }
    }
    j = j + 1;
  }
  if( j != m[0] ){
    printf("Wrong number of points found in %s.\n",argv[2]);
    exit(1);
  }
  fclose(A_file);
  /* print out the modes for plotting */
  A_file = fopen("mode_exp.dat", "w");
  for(j=0; j<m[0]; j++){
    fprintf(A_file,"%g %g ",x[j], y[j]);
    for(i=0; i<n[0]; i++){
      fprintf(A_file, "%g ",A[i][j]);
    }
    fprintf(A_file, "\n");
  }
  fclose(A_file);
  /* normalize each row, first calculate the inner product of the
     vector with itself, then divide all of the entries by the
     square root of the result */
  for(i=0; i<n[0]; i++){
    scale[0] = inner(A[i], A[i], m);
    scale[0] = sqrt(scale[0]);
    for(j=0; j<m[0]; j++){
      A[i][j] = A[i][j]/scale[0];
    }
  }
  /* Check and see if the measured modes are close to orthogonal with
     each other, then print it out for a latex table.

     The inner product of the i^th vector with the j^th vector, j !=
     i, should be close to zero */
```

```
  ortho_tab = fopen("ortho_exp_tab.tex","w");
  if(ortho_tab == NULL){
    printf("Couldn't open ortho_exp_tab.tex for write.\n");
    exit(1);
  }
  fprintf(ortho_tab, "\\begin{tabular}{|c|c|c|c|c|c|c|c|c|c} \\hline \n");
  for(i=0; i<n[0]; i++){
    for(j=0; j<n[0]; j++){
      ortho[i][j] = inner(A[i],A[j],m);
      if(j == n[0]-1){
if( ortho[i][j] < 0) fprintf(ortho_tab, " %1.2f ",ortho[i][j]);
else fprintf(ortho_tab, "  %1.2f ", ortho[i][j]);
      }else{
if( ortho[i][j] < 0) fprintf(ortho_tab, " %1.2f & ",ortho[i][j]);
else fprintf(ortho_tab, "  %1.2f & ", ortho[i][j]);
      }
    }
    fprintf(ortho_tab, "\\\\ \\hline \n");
  }
  fprintf(ortho_tab, "\\end{tabular}\n");
  fclose(ortho_tab);

  /* Of course they aren't orthogonal, so use Gramm-Schmidt to
     subtract off the non-orthogonal parts */
  for(i=1; i<n[0]; i++){/* loop over the modes */
    for(k=i; k>0; k--){/* loop over the previously orthogonalized vectors */
      scale[0] = inner(A[k-1],A[i],m);
      for(j=0; j<m[0]; j++){
B[i][j] = B[i][j] + scale[0]*A[k-1][j]; /* store the amount we
   are subtracting
   off, to look at
   latter */
A[i][j] = A[i][j] - scale[0]*A[k-1][j];
      }
    }
    /* now normalize the new vector */
    scale[0] = inner(A[i], A[i], m);
    scale[0] = sqrt(scale[0]);
    for(j=0; j<m[0]; j++){
      A[i][j] = A[i][j]/scale[0];
    }
  }
  /* print it out just to see if we did it right, should be the identity */
  ortho_tab = fopen("ortho_GS_tab.tex","w");
  if(ortho_tab == NULL){
    printf("Couldn't open ortho_GS_tab.tex for write.\n");
```

```
    exit(1);
  }
  fprintf(ortho_tab, "\\begin{tabular}{|c|c|c|c|c|c|c|c|c|c} \\hline \n");
  for(i=0; i<n[0]; i++){
    for(j=0; j<n[0]; j++){
      ortho[i][j] = inner(A[i],A[j],m);
      if(j == n[0]-1){
if( ortho[i][j] < 0) fprintf(ortho_tab, " %1.2f ",ortho[i][j]);
else fprintf(ortho_tab, "  %1.2f ", ortho[i][j]);
      }else{
if( ortho[i][j] < 0) fprintf(ortho_tab, " %1.2f & ",ortho[i][j]);
else fprintf(ortho_tab, "  %1.2f & ", ortho[i][j]);
      }
    }
    fprintf(ortho_tab, "\\\\ \\hline\n");
  }
  fprintf(ortho_tab, "\\end{tabular}\n");
  fclose(ortho_tab);

  /* print out the modes for plotting */
  A_file = fopen("mode_ortho.dat", "w");
  for(j=0; j<m[0]; j++){
    fprintf(A_file,"%g %g ",x[j], y[j]);
    for(i=0; i<n[0]; i++){
      fprintf(A_file, "%g ",A[i][j]);
    }
    fprintf(A_file, "\n");
  }
  fclose(A_file);
  /* print out the subtracted components for plotting */
  A_file = fopen("mode_diff.dat", "w");
  for(j=0; j<m[0]; j++){
    fprintf(A_file,"%g %g ",x[j], y[j]);
    for(i=0; i<n[0]; i++){
      fprintf(A_file, "%g ",B[i][j]);
    }
    fprintf(A_file, "\n");
  }
  fclose(A_file);

  /* output the file needed for the structural model */
  printf("Mass norm: %g, %d, %g\n",mass[0],m[0],sqrt(mass[0]/m[0]));
  A_file = fopen("modes.dat", "w");
  for(j=0; j < m[0]; j++){
    for(i=0; i < n[0]; i++){
      fprintf(A_file, "%1.12e ", A[i][j]/sqrt(mass[0]/m[0]) );
```

```
    }
    fprintf(A_file,"\n");
  }
  fclose(A_file);

  return(0);
}

RKIND inner(RKIND *v1,  RKIND *v2, IKIND *len)
{
  RKIND p = 0;
  int i;
  for(i=0; i<len[0]; i++){
    p = p + v1[i]*v2[i];
  }
  return(p);
}
```

Appendix C. Source for structural model time integration

The structural model is integrated in time by an explicit multistage Runge-Kutte scheme as well as a second order semi-implicit scheme. A sample input deck for the program is shown below, hash marks denote comments.

```
# mode.input
#
# An input deck for time integration of the modal equations of motion
#   generally keyword followed by value(s), can be in any order, but all of
#   them are usually required

############
## Files ##
############
# the mode file contains the orthonormal mode shapes in columns
mode-file modes.dat
# the 2D location of each of the sample points
xy-file xy.dat
# this file contains the natural frequencies squared in rad/s, the first value
# is the frequency associated with the shape in the first column, etc
spectral stiffness.dat
#################
## Parameters ##
#################
# number of modes
n 9
# number of points
m 255
# integration method
method rk

# step size
# four steps per cycle of the highest freq
#h 2.6559014129395500e-3
# ten steps per cycle of the highest freq
#h 1.06236056517582e-3
# fifteen steps per cycle of the highest freq
#h 7.08240376783881e-4
# twenty steps per cylce of the highest freq
h 5.3118028258791e-4

# total number of time steps to take
iterations 2000
# write data this many iterations
report 1
```

```
# there should be n values on the line after perturb, the integer after
# perturb is 0 for displacement, 1 for velocity
perturb 0
-1.0 0.8889 -0.7778 0.6667 -0.5556 0.4444 -0.3333 0.2222 -0.1111
```

C.1 chkmalloc.c

```
#include <stdio.h>
#include <stdlib.h>
#include "chkmalloc.h"
/* this routine taken from C programming notes by Steve Summit */

void *chkmalloc(size_t sz)
{
void *ret = malloc(sz);
if(ret == NULL)
{
fprintf(stderr, "Out of memory\n");
exit(EXIT_FAILURE);
}
return ret;
}
```

C.2 getwords.c

```
#include "getwords.h"
/* this routine taken from C programming notes by Steve Summit */

int getwords(char *line, char *words[], int maxwords)
{
char *p = line;
int nwords = 0;

while(1)
{
while(isspace(*p))
p++;

if(*p == '\0')
return nwords;

words[nwords++] = p;

while(!isspace(*p) && *p != '\0')
p++;
```

```
if(*p == '\0')
return nwords;

*p++ = '\0';

if(nwords >= maxwords)
return nwords;
}
}
```

C.3 input.c

```
#include "input.h"

#define MAXLINE 200
#define MAXWORDS 20

void alloc_input(INPUT *in)
{
  in->n = (IKIND *)chkmalloc(sizeof(IKIND));
  in->m = (IKIND *)chkmalloc(sizeof(IKIND));
  in->its = (IKIND *)chkmalloc(sizeof(IKIND));
  in->h = (RKIND *)chkmalloc(sizeof(RKIND));
  in->report = (IKIND *)chkmalloc(sizeof(IKIND));
  in->method = (IKIND *)chkmalloc(sizeof(IKIND));
  in->n[0] = 0;
  in->m[0] = 0;
  in->its[0] = 0;
  in->h[0] = 0;
}

void get_input(FILE *deck, INPUT *in, RKIND ***A, RKIND ***x, \
       RKIND ***z, RKIND ***res, RKIND **omega)
{
  char line[MAXLINE];
  int ac;
  char *av[MAXWORDS];
  IKIND i,j;
  alloc_input(in);
  j=0;
  while( fgets(line, MAXLINE, deck) != NULL ){
    if(*line == '#')
      continue;
    ac = getwords(line, av, MAXWORDS-1);
    if(ac == 0)
      continue;
```

```
    if( strcmp(av[0], "mode-file") == 0 ){
      in->mode_file = fopen( av[1],"r" );
      if(in->mode_file == NULL) file_error( av[1] );
    }
    if( strcmp(av[0], "xy-file") == 0){
      in->xy_file = fopen( av[1], "r");
      if(in->xy_file == NULL) file_error( av[1] );
    }
    if( strcmp( av[0], "spectral") == 0){
      in->spectral = fopen( av[1], "r");
      if(in->spectral == NULL) file_error( av[1] );
    }
    if( strcmp( av[0], "n") == 0){
      in->n[0] = atoi( av[1] );
    }
    if( strcmp( av[0], "m") == 0){
      in->m[0] = atoi( av[1] );
    }
    if( strcmp( av[0], "h") == 0){
      in->h[0] = atof( av[1] );
    }
    if( strcmp( av[0], "iterations") == 0){
      in->its[0] = atof( av[1] );
    }
    if( strcmp( av[0], "report") == 0){
      in->report[0] = atoi( av[1]);
    }
    if( strcmp( av[0], "method") == 0){
      if( strcmp( av[1], "rk" ) == 0 ){
in->method[0] = 0;
      }
      if(strcmp( av[1], "semi-implicit") == 0){
in->method[0] = 1;
      }
      if(strcmp( av[1], "third" ) == 0 ){
in->method[0] = 2;
      }
      if(strcmp( av[1], "fourth" ) == 0 ){
in->method[0] = 3;
      }
      if(strcmp( av[1], "fifth" ) == 0 ){
in->method[0] = 4;
      }
      if(strcmp( av[1], "sixth" ) == 0 ){
in->method[0] = 5;
      }
```

```
      if(strcmp( av[1], "eighth" ) == 0 ){
in->method[0] = 6;
      }
      if(strcmp( av[1], "coupled-implicit-2" ) == 0 ){
in->method[0] = 7;
      }
      if(strcmp( av[1], "coupled-implicit-3" ) == 0 ){
in->method[0] = 8;
      }
      if(strcmp( av[1], "coupled-implicit-4" ) == 0 ){
in->method[0] = 9;
      }
      if(strcmp( av[1], "newmark" ) == 0 ){
in->method[0] = 10;
      }
      if(strcmp( av[1], "var-accel-2" ) == 0 ){
in->method[0] = 11;
      }
      if(strcmp( av[1], "rk-3" ) == 0 ){
in->method[0] = 12;
      }
    }
    if( strcmp( av[0], "perturb") == 0){
      if(in->n[0] == 0){
printf("Declare number of modes before perturbations\n");
exit(1);
      }
      fgets(line, MAXLINE, deck);
      ac = getwords(line, av, MAXWORDS-1);
      if( ac != in->n[0] ){
printf("Wrong number of perturbation values specified\n");
exit(1);
      }
      in->perturb = (RKIND *)chkmalloc(in->n[0]*sizeof(RKIND));
      for(i=0;i<in->n[0];i++) in->perturb[i] = atof( av[i] );
    }
    j++;
  }
  /* done parsing the input deck */
  if( (in->mode_file == NULL) || (in->xy_file == NULL) ){
    printf("Must specify mode file and grid file in input deck\n");
    exit(1);
  }
  (*A) = (RKIND **)chkmalloc( in->n[0]*sizeof(RKIND) );
  for(i=0; i < in->n[0]; i++)
    (*A)[i] = (RKIND *)chkmalloc( in->m[0]*sizeof(RKIND) );
```

```
  (*x) = (RKIND **)chkmalloc( DIM*sizeof(RKIND) );
  for(i=0; i < DIM; i++)
    (*x)[i] = (RKIND *)chkmalloc( in->m[0]*sizeof(RKIND) );
  (*omega) = (RKIND *)chkmalloc( in->n[0]*sizeof(RKIND) );
  (*z) = (RKIND **)chkmalloc( HIST*sizeof(RKIND) );
  (*res) = (RKIND **)chkmalloc( 3*sizeof(RKIND) );
  for(i=0; i < HIST; i++){
    (*z)[i] = (RKIND *)chkmalloc( in->n[0]*sizeof(RKIND) );
    (*res)[i] = (RKIND *)chkmalloc( in->n[0]*sizeof(RKIND) );
  }
  /* read the grid coordinates */
  i = 0;
  while( fgets(line, MAXLINE, in->xy_file) != NULL ){
    ac = getwords(line, av, MAXWORDS-1);
    if(i<in->m[0]){
      (*x)[0][i] = atof( av[0] );
      (*x)[1][i] = atof( av[1] );
    }else{
      printf("Too many points in grid file\n");
      exit(1);
    }
    i++;
  }
  j = 0;
  while( fgets(line, MAXLINE, in->mode_file) != NULL){
    ac = getwords(line, av, MAXWORDS-1);
    if(ac == in->n[0]){
      for(i=0;i<in->n[0];i++){
(*A)[i][j] = atof( av[i] );
      }
    }else{ printf("Wrong number of points on the line\n"); exit(1); }
    j++;
  }
  j = 0;
  while( fgets(line, MAXLINE, in->spectral) != NULL){
    ac = getwords(line, av, MAXWORDS-1);
    if(j< in->n[0] )
      (*omega)[j] = atof( av[0] );
    else{
      printf("Wrong number of natural frequencies given\n");
      exit(1);
    }
    j++;
  }
}
```

```
void file_error(char *filename)
{
  printf("Error with IO attempt to %s\n", filename);
  exit(1);
}
```

C.4 modal-eom.c

```
#include "modal-eom.h"

#define MAXLINE 200
#define MAXWORDS 20

int main(int argc, char *argv[])
{/* orthonormalize the modes before using them in this program */
  FILE *in_file;
  FILE *response;
  FILE *mode_out;
  IKIND j,r;
  RKIND **A;
  RKIND **x; /* x[0] is x coordinate, x[1] is y */
  RKIND **z; /* modal displacements and accelerations, or displacement
history */
  RKIND *d;
  RKIND **res; /* residual */
  RKIND *omega; /* natural frequencies, rad/s */
  IKIND *n; /* rows */
  IKIND *m; /* columns */
  RKIND *sc; /* scale factor */
  RKIND *h; /* time step size */
  RKIND *its; /* number of time steps */
  char *b1;
  b1 = (char *)chkmalloc(MAXLINE*sizeof(char));
  sc = (RKIND *)chkmalloc(sizeof(RKIND));
  h = (RKIND *)chkmalloc(sizeof(RKIND));
  its = (RKIND *)chkmalloc(sizeof(RKIND));
  n = (IKIND *)chkmalloc(sizeof(IKIND));
  m = (IKIND *)chkmalloc(sizeof(IKIND));
  mode_out = fopen("mode_hist.dat","w");
  if(argc!=2){printf("Usage:\n      %s input_file_name \n",argv[0]); exit(1);}
  else{
    in_file = fopen(argv[1],"r");
    if(in_file==NULL) file_error( argv[1] );
  }
  INPUT *in;
```

```
in = (INPUT *)chkmalloc( sizeof(INPUT) );

get_input( in_file, in, &A, &x, &z, &res, &omega);
d = (RKIND *)chkmalloc(in->m[0]*sizeof(RKIND));
perturb(in, z, omega);
j=0;r=0;
for(j=0; j<in->n[0];j++){
  printf("%g ",omega[j]);
}
printf("\n");
j=0;
while( j < in->its[0] ){/* time loop */
  /* write out the shapes only every report iterations */
  if( j % in->report[0] == 0){
    sprintf(b1, "response-"IFORM".out", r);
    response = fopen(b1, "w");
    write_shapes(in, response, A, x, z, d);
    fclose(response);
    r++;
  }
  /* write the modal amplitudes every iteration */
  write_modes(in, mode_out, z);
  /* integrate */
  if( in->method[0] == 0 )
    rk( in, z, omega, res );
  if( in->method[0] == 1 )
    semi_implicit( in, z, omega );
  if( in->method[0] == 2 )
    third_order( in, z, omega );
  if( in->method[0] == 3 )
    fourth_order( in, z, omega );
  if( in->method[0] == 4 )
    fifth_order( in, z, omega );
  if( in->method[0] == 5 )
    sixth_order( in, z, omega );
  if( in->method[0] == 6 )
    eighth_order( in, z, omega );
  if( in->method[0] == 7 )
    coupled_imp_2( in, z, omega );
  if( in->method[0] == 8 )
    coupled_imp_3( in, z, omega );
  if( in->method[0] == 9 )
    coupled_imp_4( in, z, omega );
  if( in->method[0] == 10 )
    newmark( in, z, omega );
  if( in->method[0] == 11 )
```

```
      coupled_var_accel_2( in, z, omega );
    if( in->method[0] == 11 )
      rk_3( in, z, omega, res );

    j++;
  }
  return(0);
}

void write_shapes(INPUT *in, FILE *f, RKIND **A, RKIND **x, RKIND **z, RKIND *d)
{
  IKIND i,j;
  for(j=0; j < in->m[0]; j++){
    d[j] = 0;
    for(i=0;i<in->n[0];i++){
      d[j] = d[j] + A[i][j]*z[0][i];
    }
    fprintf(f, RFORM" "RFORM" "RFORM" \n", x[0][j], x[1][j], d[j]);
  }
}

void write_modes(INPUT *in, FILE *f, RKIND **z)
{
  IKIND i;
  for(i=0; i < in->n[0]; i++) fprintf(f, RFORM" ",z[0][i]);
  fprintf(f,"\n");
}
```

C.5 solve.c

```
#include "solve.h"

void perturb(INPUT *in, RKIND **z, RKIND *omega)
{
  IKIND i;
  for(i=0; i < in->n[0]; i++){
    z[0][i] = in->perturb[i];
    if( in->method[0] == 0 ){/* rk */
      z[1][i] = ZERO;
    }else{
      if( in->method[0] == 7 ){/* coupled implicit */
z[1][i] = in->perturb[i];
z[2][i] = ZERO;
z[3][i] = ZERO;
      }else{
if( in->method[0] == 10 ){/* newmark */
```

```
  z[1][i] = in->perturb[i];
  z[2][i] = ZERO;
  z[3][i] = ZERO;
  z[4][i] = ZERO;
  z[5][i] = ZERO;
}else{
  if(in->method[0] == 12 ){/* rk scheme with three eqns */
    z[2][i] = ZERO;
    z[4][i] = ZERO;
  }else{/* one of the finite difference methods */
    z[1][i] = in->perturb[i];
    z[2][i] = in->perturb[i];
    z[3][i] = in->perturb[i];
    z[4][i] = in->perturb[i];
    z[5][i] = in->perturb[i];
    z[6][i] = in->perturb[i];
    z[7][i] = in->perturb[i];
    z[8][i] = in->perturb[i];
    z[9][i] = in->perturb[i];
  }
}
      }
    }
  }
}

void rhs(INPUT *in, RKIND **z, RKIND *omega, RKIND **res)
{/* z[0] is displacements, z[1] is velocitiess */
  IKIND j;
  for(j=0; j < in->n[0]; j++){
      res[0][j] = z[1][j];
      res[1][j] = -omega[j]*z[0][j];
  }
}

void rk(INPUT *in, RKIND **z, RKIND *omega, RKIND **res)
{
  IKIND i,j,s;
  RKIND *alpha;
  IKIND stages = 4;
  RKIND **z1;
  z1 = (RKIND **)chkmalloc(2*sizeof(RKIND));
  z1[0] = (RKIND *)chkmalloc(in->n[0]*sizeof(RKIND));
  z1[1] = (RKIND *)chkmalloc(in->n[0]*sizeof(RKIND));
  alpha = (RKIND *)chkmalloc(stages*sizeof(RKIND));
  alpha[0] = ONE/FOUR;
```

```
  alpha[1] = ONE/THREE;
  alpha[2] = ONE/TWO;
  alpha[3] = ONE;

  /* store the old data */
  for(j=0; j < 2; j++){
    for(i=0; i < in->n[0]; i++){
      z1[j][i] = z[j][i];
    }
  }
  for(s=0; s<stages; s++){

    rhs( in, z, omega, res);

    for(j=0; j < 2; j++){
      for(i=0; i < in->n[0]; i++){

z[j][i] = z1[j][i] - alpha[s]*res[j][i]*in->h[0];

      }
    }
  }
}

void newmark(INPUT *in, RKIND **z, RKIND *omega)
{/* z: 0, 1 displacement
       2, 3 velocity
       4, 5 acceleration */
  RKIND dt;
  IKIND i;
  dt = in->h[0];
  for( i=0; i<in->n[0]; i++ ){
    /* slide the solution back */
    z[5][i] = z[4][i];
    z[3][i] = z[2][i];
    z[1][i] = z[0][i];
    /* update acceleration */
    z[4][i] = -omega[i]*( z[0][i] + dt*z[2][i] + dt*dt*z[5][i]/3 )/
      ( 1 + dt*dt*omega[i]/6 );
    /* update velocity */
    z[2][i] = z[3][i] + dt*( z[5][i] + z[4][i] )/TWO;
    /* update displacement */
    z[0][i] = z[1][i] + dt*z[3][i] + dt*dt*( TWO*z[5][i] + z[4][i] )/SIX;
  }
}
```

```
void coupled_var_accel_2(INPUT *in, RKIND **z, RKIND *omega)
{/*z: 0,1 displacement
      2,3 velocity
      4,5 acceleration */
  RKIND a, b1, b2, b3, dt;
  IKIND i;
  dt = in->h[0];
  for(i=0; i<in->n[0]; i++){
    a = FOUR*dt*dt*omega[i] + 9.0;
    /* right hand side vector */
    b1 = FOUR*z[0][i] - z[1][i];
    b2 = FOUR*z[2][i] - z[3][i];
    b3 = FOUR*z[4][i] - z[5][i];
    /* move the solution back */
    z[1][i] = z[0][i];
    z[3][i] = z[2][i];
    z[5][i] = z[4][i];
    /* calculate the new solution */
    z[0][i] = b1/THREE + TWO*dt*b2/a + FOUR*dt*dt*b3/(3*a);
    z[2][i] = THREE*b2/a + TWO*dt*b3/a;
    z[4][i] = -TWO*dt*omega[i]*b2/a + THREE*b3/a;
  }
}

void semi_implicit(INPUT *in, RKIND **z, RKIND *omega)
{
  IKIND i;
  /* move everything back one */
  for(i=0; i<in->n[0]; i++){
    z[3][i] = z[2][i];
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
  }
  for( i=0; i< in->n[0]; i++ ){/* loop over the modes */
    z[0][i] = ( FIVE*z[1][i] - FOUR*z[2][i] + z[3][i] )/( TWO + pow(in->h[0],TWO)*omega[i] )
  }
}

void third_order(INPUT *in, RKIND **z, RKIND *omega)
{/*XXX unstable XXX*/
  IKIND i;
  RKIND a;
  /* move everything back one */
  for(i=0; i<in->n[0]; i++){
    z[4][i] = z[3][i];
    z[3][i] = z[2][i];
```

```
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
  }
  for( i=0; i< in->n[0]; i++ ){/* loop over the modes */
    a = ( 35.0 + 12.0*pow(in->h[0],TWO)*omega[i] );
    z[0][i] = ( 104.0*z[1][i] - 114.0*z[2][i] + 56.0*z[3][i] -11.0*z[4][i] )
      / a;
  }
}

void fourth_order(INPUT *in, RKIND **z, RKIND *omega)
{/*XXX unstable XXX*/
  IKIND i;
  RKIND a;
  /* move everything back one */
  for(i=0; i<in->n[0]; i++){
    z[5][i] = z[4][i];
    z[4][i] = z[3][i];
    z[3][i] = z[2][i];
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
  }

  for( i=0; i< in->n[0]; i++ ){/* loop over the modes */
    a = ( 45.0 + 12.0*pow(in->h[0],TWO)*omega[i] );
    z[0][i] = ( 154.0*z[1][i] - 214.0*z[2][i] + 156.0*z[3][i] - 61.0*z[4][i] + 10.0*z[5][i]
  }
}

void fifth_order(INPUT *in, RKIND **z, RKIND *omega)
{/*XXX this one is unstable XXX*/
  IKIND i;
  RKIND a;
  /* move everything back one */
  for(i=0; i<in->n[0]; i++){
    z[6][i] = z[5][i];
    z[5][i] = z[4][i];
    z[4][i] = z[3][i];
    z[3][i] = z[2][i];
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
  }
  for( i=0; i< in->n[0]; i++ ){/* loop over the modes */
    a = ( 203/45 + pow(in->h[0],TWO)*omega[i] );
    z[0][i] = ( 87*z[1][i]/5 - 117*z[2][1]/4 + 254*z[3][i]/9 - 33*z[4][i]/2
+ 27*z[5][i]/5 - 137*z[6][i]/180 )/a;
```

```
  }
}

void sixth_order(INPUT *in, RKIND **z, RKIND *omega)
{
  IKIND i;
  RKIND a;
  /* move everything back one */
  for(i=0; i<in->n[0]; i++){
    z[7][i] = z[6][i];
    z[6][i] = z[5][i];
    z[5][i] = z[4][i];
    z[4][i] = z[3][i];
    z[3][i] = z[2][i];
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
  }
  for( i=0; i< in->n[0]; i++ ){/* loop over the modes */
    a = ( 938 + 180*pow(in->h[0],TWO)*omega[i] );
    z[0][i] = ( 4014*z[1][i] - 7911*z[2][i] + 9490*z[3][i] - 7380*z[4][i]
+ 3618*z[5][i] - 1019*z[6][i] + 126*z[7][i] )/a;
  }
}

void eighth_order(INPUT *in, RKIND **z, RKIND *omega)
{/*XXX unstable XXX*/
  IKIND i;
  RKIND a;
  /* move everything back one */
  for(i=0; i<in->n[0]; i++){
    z[9][i] = z[8][i];
    z[8][i] = z[7][i];
    z[7][i] = z[6][i];
    z[6][i] = z[5][i];
    z[5][i] = z[4][i];
    z[4][i] = z[3][i];
    z[3][i] = z[2][i];
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
  }
  for( i=0; i< in->n[0]; i++ ){/* loop over the modes */
    a = ( 32575 + 5040*pow(in->h[0],TWO)*omega[i] );
    z[0][i] = ( 165924*z[1][i] - 422568*z[2][i] + 704368*z[3][i]
- 818874*z[4][i] + 667800*z[5][i] - 375704*z[6][i]
+ 139248*z[7][i] - 30223*z[8][i] + 3044*z[9][i] )/a;
  }
```

```
}

void coupled_imp_2(INPUT *in, RKIND **z, RKIND *omega)
{/*z 0, 1 : displacement
     2, 3 : velocity */
  IKIND i;
  RKIND a, b1, b2;
  for(i=0; i <in->n[0]; i++){/* loop over the modes */
    a = FOUR*in->h[0]*in->h[0]*omega[i] + 9;
    b1 = FOUR*z[0][i] - z[1][i];
    b2 = FOUR*z[2][i] - z[3][i];
    /* move the velocity back one */
    z[3][i] = z[2][i];
    /* modve the displacements back one */
    z[1][i] = z[0][i];
    /* calculate the new ones */
    z[0][i] = ( THREE*b1 + 2*in->h[0]*b2 )/a;
    z[2][i] = ( -TWO*in->h[0]*omega[i]*b1 + THREE*b2 )/a;
  }
}

void coupled_imp_3(INPUT *in, RKIND **z, RKIND *omega)
{/*XXX unstable XXX*/
/*z 0, 1, 2 : displacement
    3, 4, 5 : velocity */
  IKIND i;
  RKIND a, b1, b2;
  for(i=0; i <in->n[0]; i++){/* loop over the modes */
    a = 36.0*in->h[0]*in->h[0]*omega[i] + 121.0;
    b1 = 18.0*z[0][i] - 9.0*z[1][i] + 2.0*z[2][i];
    b2 = 18.0*z[3][i] - 9.0*z[4][i] + 2.0*z[5][i];
    /* move the displacements back one */
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
    /* move the velocities back one */
    z[5][i] = z[4][i];
    z[4][i] = z[3][i];
    /* calculate the new ones */
    z[0][i] = (11.0*b1 + 6.0*in->h[0]*b2)/a;
    z[3][i] = (-6.0*in->h[0]*omega[i]*b1 + 11.0*b2)/a;
  }
}

void coupled_imp_4(INPUT *in, RKIND **z, RKIND *omega)
{/*XXX unstable XXX*/
/*z 0, 1, 2, 3 : displacement
```

```
    4, 5, 6, 7 : velocity */
  IKIND i;
  RKIND a, b1, b2;
  for(i=0; i <in->n[0]; i++){/* loop over the modes */
    a = 144*in->h[0]*in->h[0]*omega[i] + 625;
    b1 = 48*z[0][i] - 36*z[1][i] + 16*z[2][i] -3*z[3][i];
    b2 = 48*z[4][i] - 36*z[5][i] + 16*z[6][i] -3*z[7][i];
    /* move the displacements back one */
    z[3][i] = z[2][i];
    z[2][i] = z[1][i];
    z[1][i] = z[0][i];
    /* move the velocities back one */
    z[7][i] = z[6][i];
    z[6][i] = z[5][i];
    z[5][i] = z[4][i];
    /* calculate the new ones */
    z[0][i] = (25*b1 + 12*in->h[0]*b2)/a;
    z[4][i] = (-12*in->h[0]*omega[i]*b1 + 25*b2)/a;
  }
}

void rk_3(INPUT *in, RKIND **z, RKIND *omega, RKIND **res)
{/*z: 0,1 displacement
      2,3 velocity
      4,5 acceleration */
  IKIND i,j;
  RKIND dt;
  RKIND *alpha;
  IKIND stages = 4;
  alpha = (RKIND *)chkmalloc(stages*sizeof(RKIND));

  dt = in->h[0];
  alpha[0] = ONE/FOUR;
  alpha[1] = ONE/THREE;
  alpha[2] = ONE/TWO;
  alpha[3] = ONE;
  /* store the old solution */
  for(i=0;i<in->n[0];i++){
    z[1][i] = z[0][i];
    z[3][i] = z[2][i];
    z[5][i] = z[4][i];
  }
  for(j=0; j < stages; j++){
    for(i=0; i < in->n[0]; i++){
      res[0][i] = z[2][i];
      res[1][i] = z[4][i];
```

```
      res[2][i] = -omega[i]*z[2][i];
      z[0][i] = z[1][i] - alpha[j]*dt*res[0][i];
      z[2][i] = z[3][i] - alpha[j]*dt*res[1][i];
      z[4][i] = z[5][i] - alpha[j]*dt*res[2][i];
    }
  }
}
```

This routine uses the *fftw* library released by MIT under the GNU General Public License, and available at `http://www.fftw.org`.

D.1 cl-fft.c

```
/* This is a simple program that reads a column of data from a text
   file and performs a discrete fourier transform on it, normalizes the
   output by the DC offset, (if it isn't zero), outputs the result to
   another text file */

#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <fftw3.h>
#include "getwords.h"

#define MAXLINE 200
#define MAXWORDS 2

int main( int argc, char *argv[])
{
  double *f, dt, norm;
  double *in;
  fftw_complex *out;
  fftw_plan p;
  int n, ac, i;
  FILE *in_f;
  FILE *out_f;
  char line[MAXLINE];
  char *av[MAXWORDS];
  if( argc != 3 ){
    printf("Error.\nUsage:\n"
   "  %s SAMPLE_RATE datafile\n",argv[0]);
    exit(1);
  }
  dt = atof( argv[1] );
  in_f = fopen( argv[2], "r" );
  sprintf( line, "%s.fft", argv[2] );
  out_f = fopen(line, "w");
  if( in_f == NULL ){
    printf("Error\nCould not open\n  %s\nfor read.\n",argv[1]);
    exit(1);
  }
  n = 0;
  while( fgets(line, MAXLINE, in_f ) != NULL ){
```

```
    n++;
  }
  if( n == 0 ){
    printf("Error.\nNo lines of data.\n");
    exit(1);
  }
  fclose(in_f);
  in = (double *)malloc( sizeof(double)*n );
  f = (double *)malloc( sizeof(double)*n/2 );
  out = (fftw_complex *)fftw_malloc( sizeof(fftw_complex)*(1+n/2) );
  in_f = fopen( argv[2], "r");
  n = 0;
  while( fgets(line, MAXLINE, in_f ) != NULL ){
    ac = getwords(line, av, MAXWORDS-1 );
    in[n] = atof( av[0] );
    n++;
  }

  p = fftw_plan_dft_r2c_1d( n, in, out, FFTW_ESTIMATE );

  fftw_execute(p);

  for(i=0;i<n/2;i++){
    /* frequency real imaginary magnitude */
    if(i==0){
      if( out[i][0] != 0 )
norm = out[i][0];
      else
norm = 1.0;
      printf("Average: %g\n", norm/n);
    }
    in[i] = sqrt( out[i][0]*out[i][0] + out[i][1]*out[i][1] );
    f[i] = ((double)i)/( dt*(double)n );
    fprintf(out_f, "%g %g %g %g\n", f[i], out[i][0]/norm, out[i][1]/norm,
    in[i]/norm );
  }

  fftw_destroy_plan(p);
  fftw_free(in);  fftw_free(out);

  return(0);
}
```

And now for something completely different ...

Sonnet of a MAV

```
We know flapping wings, and flopping wings,
That ought drive us to distraction.
Isn't it always the membrane things,
That create the main attraction?

Their carbon fiber ribs and spars
With nylon in betwixt
Refuse to bend or stretch too far
With fluids in the mix.

And towards the middle of the night,
When most folks are gone to sleep,
We let our bits in machines take flight,
So we know those to throw away or keep.

But in the last analysis our derivations were in vain.
The world, it is non-linear! Now isn't that a pain?
```

Vita

Joshua Allen Stults graduated from Virgil I. Grissom High School in Huntsville, Alabama, and then attended the United States Air Force Academy in Colorado Springs, Colorado. He was commissioned as a second lieutenant, and earned his Bachelors of Science in Aeronautical Engineering on 28 May 2003. Immediately after graduation from the United States Air Force Academy he was assigned to the Air Force Institute of Technology where he earned the degree of Master of Science in Aeronautical Engineering in March 2005. Upon graduation he was assigned to the 46^{th} Operations Group, 46^{th} Test Wing, Eglin Air Force Base, Florida. In April 2005 he married his high-school sweetheart.

Joshua is an active member of the American Institute of Aeronautics and Astronautics as well as the Tau Beta Pi Engineering Honor Society, and the Sigma Gamma Tau National Aerospace Engineering Honor Society. His research interests focus on applied computational aerodynamics and mechanics, especially fluid/structure interaction.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2005	Master's Thesis	Sep 2003 — Mar 2005

4. TITLE AND SUBTITLE

Computational Aeroelastic Analysis of Micro Air Vehicle with Experimentally Determined Modes

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Stults, Joshua A., 2LT, USAF

5d. PROJECT NUMBER

2004-095

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GAE/ENY/05-M23

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/MNAV
Capt Ian Bautista
Eglin AFB, FL 32542

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

A computational aeroelastic analysis of a micro air vehicle (MAV) is conducted. This MAV has a 24 inch wing span, and is designed for local area reconnaissance. Wind tunnel data for the MAV with a rigid carbon fiber wing and a flexible carbon fiber ribbed nylon wing are compared to CFD results incorporating static and dynamic deformations. We use laser vibrometry to determine the mode shapes of the flexible wing. From these shapes, CFD grid deformations are calculated as part of a closely-coupled aeroelastic solution method. The accuracy of MAV performance predictions using CFD with and without aeroelastic modeling is evaluated against previous wind-tunnel experiments. The performance benefits of the flexible wing, and the applicability and limitations of the model are evaluated in the present research effort. Some suggestions are made as to improvements that can be made to increase the range of applicability and the accuracy of the model.

15. SUBJECT TERMS

Micro Air Vehicles, Fluid Structure Interaction, Aeroelasticity, Laser Vibrometry, Incompressible Flow, Modal Analysis

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Raymond C. Maple, LtCol, USAF (ENY)
U	U	U	UU	95	19b. TELEPHONE NUMBER *(include area code)* (937) 255–3636, ext 4577